Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет»
(национальный исследовательский университет)
Факультет математики, механики и компьютерных технологий
Кафедра прикладной математики и программирования

РАБОТА ПРОВЕРЕНА
Рецензент зав. уч. каб. ЦПЛ ММКТ
_____ / А.А. Эбель /
«14» 07. 20___ г.

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой, д.ф.-м.н.,
доцент
_____ А.А. Замышляева
«18» 07 2016 г.

Разработка модуля «Рабочее место крановщика» для транспортно – складской системы CS-W на ОАО «ЧТПЗ» цех «Высота 239»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ ЮУрГУ–231000.2016.079.ПЗ ВКР

Руководитель работы,
доцент к.т.н.
_____ /Б.М. Кувшинов
«____» _____ 2016 г.
Автор работы
Студент группы ММиКН-474
_____ / Д.В. Лелеков
«12» 07 2016 г.
Нормоконтролер,
доцент к.ф.-м.н.
_____ /С.У. Турлакова
«12» 07. 2016 г.

Челябинск 2016

Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет»
(национальный исследовательский университет)
Факультет математики, механики и компьютерных наук
Кафедра прикладной математики
Направление подготовки Программная инженерия

УТВЕРЖДАЮ
Заведующий кафедрой
_____/ Л.А. Прокудина
_____2015 г.

З А Д А Н И Е

на выпускную квалификационную работу студента
Лелекова Дениса Валерьевича
Группа ММиКН-474

1. **Тема работы** Разработка модуля «Рабочее место крановщика» для транспортно – складской системы CS-W на предприятии ОАО «ЧТПЗ» цех «Высота 239»

   утверждена приказом по университету от « 15 » 04 2016 г. № 661

2. **Срок сдачи студентом законченной работы** « 12» **июля 2016 г.**

3. **Исходные данные к работе**
   3.1. СУБД MySQL.
   3.2. Среда программирования – Microsoft Visual Studio, DevExpress.
   3.3. Использование на предприятии определенного количества кранов, а также выполнение определенных типов заданий.

Перечень вопросов, подлежащих разработке
4.1. Отражение состояния склада.
4.2. Отражение состояния оборудования.
4.3. Обзор состояния кранов.
4.4. Обзор рабочих заданий.
4.5. Разработка структуры программы.
4.6. Разработка интерфейса.
4.7. Разработка модуля «Рабочее место крановщика».
4.8. Тестирование и отладка программного продукта.
4.9. Разработка программной документации(описание программы, текст программы, руководство пользователя)

4. **Иллюстративный материал** (плакаты, альбомы, раздаточный материал, макеты, электронные носители и др.)
   5.1. Структура модуля (диаграмма классов). Демонстрационный плакат – 1 л.
   5.2. Мультимедийная презентация – 14 слайдов.
   Общее количество иллюстраций – 29.
5. **Дата выдачи задания** «1» октября 2015 г.

Руководитель_____ /Б.М. Кувшинов/
(подпись)

Задание принял к исполнению_____ /Д.В. Лелеков/
(подпись)

6. **Календарный план**

Наименование этапов выпускной квалификационной работы	Срок выполнения этапов работы	Отметка о выполнении руководителя
1. Отражение состояния склада	01.10.15 – 07.11.15	
2. Отражение состояния оборудования	09.11.15– 19.12.15	
3. Обзор состояния кранов	21.12.15-06.02.16	
4. Обзор рабочих заданий	08.02.16 – 05.03.16	
5. Разработка структуры программы	07.03.16 – 26.03.16	
6. Разработка интерфейса	28.03.16 – 16.04.16	
7. Разработка модуля «Рабочее место крановщика»	28.03.16 – 04.04.16	
8. Тестирование и отладка программного продукта	05.04.16 – 16.04.16	
9. Подготовка пояснительной записки дипломной работы	18.04.16 – 01.05.16	
10. Разработка программной документации	02.05.16 – 15.05.16	
11. Проверка работы руководителем, исправление замечаний	16.05.16 – 28.05.16	
12. Нормоконтроль	29.05.16 – 11.06.16	
13. Подготовка иллюстративного материала и доклада	12.06.16 – 30.06.16	
14. Рецензирование, представление зав. Кафедрой	01.07.16 – 12.07.16	

Заведующий кафедрой _____ /Л.А. Прокудина/
(подпись)

Руководитель работы _____ /Б.М. Кувшинов/
(подпись)

Студент _____ / Д.В. Лелеков /

АННОТАЦИЯ

Лелеков Д.В. Разработка модуля «Рабочее место крановщика» для транспортно – складской системы CS-W на предприятии ОАО «ЧТПЗ» цех «Высота 239». – Челябинск: ЮУрГУ, ММиКН-474, 105 с., 29 ил., 1 табл., библиогр. список – 19 наим., 3 прил., 1 л. плакатов ф. А1.

В выпускной квалификационной работе реализован модуль «Рабочее место крановщика». Изучена структура всей системы управления производством. Разработан интерфейс и структура модуля «Рабочее место крановщика». Модуль интегрирован в общую систему управления производством. Проведено тестирование как отдельно модуля, так и его работы во всей системе управления производством.

В приложении приведен текст программы, реализующий модуль «Рабочее место крановщика» на языке C#, а также инструкция пользователю.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

С момента появления первых ремесленных предприятий человек старается ускорить, улучшить и автоматизировать выполняемую им работу. Осознавая данные идеи, люди, создавали различные механизмы, изобретали удивительные вещи. По средствам данных идей, шел прогресс, который привел человечество к тому, что оно имеет. Но и сейчас, когда высокие технологии получили повсеместное применение, все также остро стоит задача ускорения, оптимизации и автоматизации деятельности предприятия.

Для решения подобных задач в нынешнее время существуют универсальные программные комплексы. Мощные системы, позволяющие провести тонкую настройку их работы и имеющие большое количество инструментов. Основными плюсами таких программных продуктов являются их универсальность применения, обилие технической документации и развитая система поддержки. К минусам же можно отнести довольно высокую стоимость таких систем, сложность интеграции в уже существующую инфраструктуру предприятия, но кроме того потребуется ещё и наличие специалистов на предприятии, способных провести нужную настройку таких систем.

С учетом современных требований по автоматизации всех этапов жизненного цикла изделия, включающего и процесс производства, в управлении современными предприятиями широко используются автоматизированные системы управления и поддержки принятия решений. Эти системы обычно классифицируют по выполняемым функциям. В частности весьма востребованными являются производственные исполнительные MES – системы, основным назначением которых является автоматизация планирования и управления производством.

Цель работы – разработка модуля «Рабочее место крановщика» для транспортно – складской системы CS – W на предприятии ОАО «ЧТПЗ» цех «Высота 239».

Задачи:

- Изучить структуру всей системы управления производством
- Спроектировать структуру модуля «Рабочее место крановщика»
- Разработать интерфейс модуля «Рабочее место крановщика»
- Интегрировать модуль в систему управления производством

Объект исследования - Транспортно складская система CS-W на ОАО «Челябинский трубопрокатный завод», цех «Высота 239».

В первой главе выпускной квалификационной работы приведен обзор существующих подходов к разработке модуля, а также основные системы управления производством.

Во второй главе показана структура и описание модуля «Рабочее место крановщика».

В третьей главе выпускной квалификационной работы приведена реализация данного модуля и его тестирование.

1. ОБЗОР СУЩЕСТВУЮЩИХ ПОДХОДОВ

1.1 Информационно-управляющая структура производственного предприятия

Автоматизация решает задачи планирования и управления промышленными предприятиями. В условиях жесткой конкуренции, динамического рынка даже самые консервативные или небогатые предприятия не могут позволить себе отказаться от столь мощного средства эволюции, как автоматизация.

Теоретические попытки классифицировать решаемые задачи привели к разделению их на две группы:

- Технологические, производственные
- Экономические, административные и логистические.

Первая относится строго к производственной деятельности предприятия, вторая – к административно-хозяйственной. В целом, информационно-управляющую структуру предприятия можно представить в виде пирамиды, представленной на (рисунок 1.1).

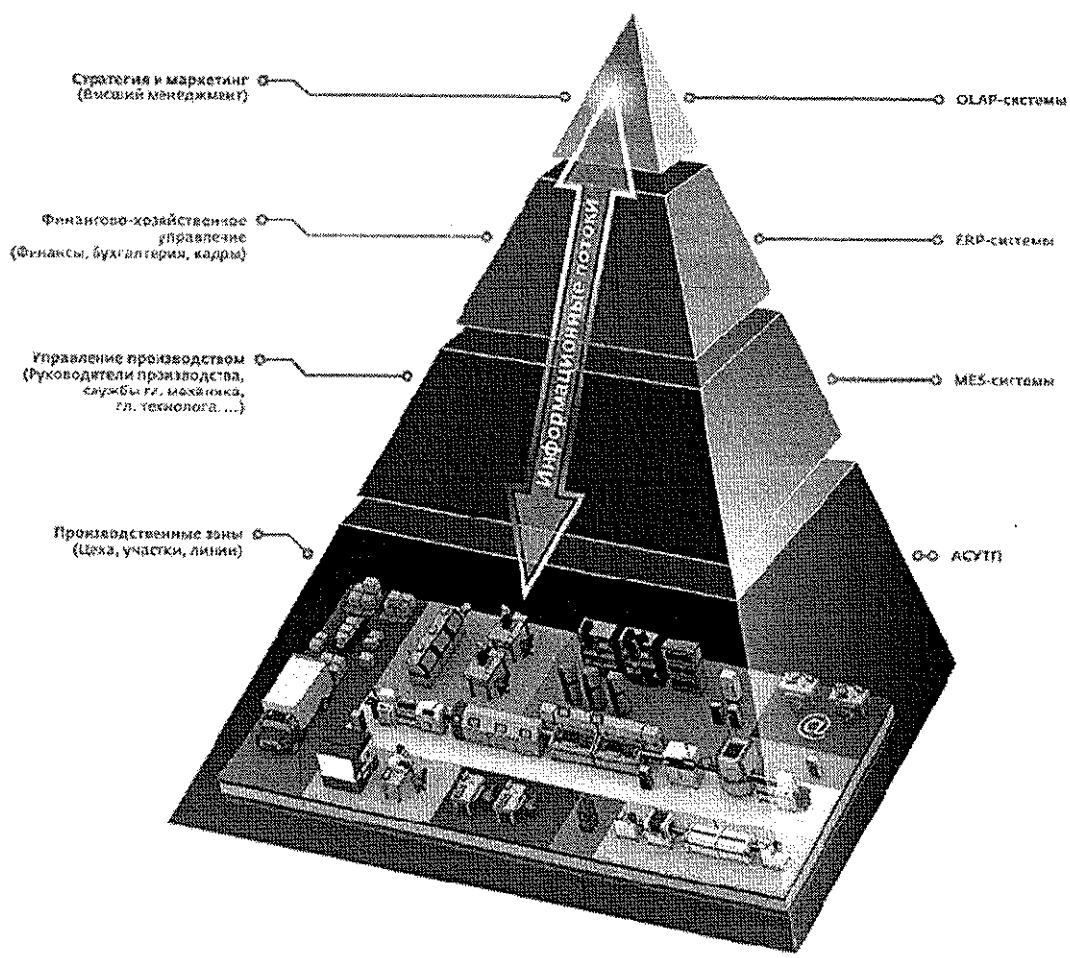


Рисунок 1.1

Представленная выше пирамида дает схематическое представление о структуре информационной системы автоматизации промышленного предприятия, разделенной на четыре уровня:

- Первый уровень – АСУТП
- Второй уровень – MES – система
- Третий уровень – ERP – система
- Четвертый уровень – OLAP – система

АСУТП – автоматизированные системы управления технологическими процессами.

MES – (Manufacturing Execution System) - исполнительная система производства, информационно-вычислительная система. Системы такого класса решают задачи синхронизации, координируют, анализируют и оптимизируют выпуск продукции в рамках какого-либо производства в режиме реального времени.

ERP – (Enterprise Resource Planning) – система планирования ресурсов предприятия. Основное назначение ERP – управление финансовой и хозяйственной деятельности предприятия . ERP – система работает на самом верхнем уровне в иерархической лестнице систем управления, она затрагивает основные аспекты всех элементов производственной и торговой деятельности предприятия.

OLAP – (On-Line Analytic Processing) – оперативный многомерный анализ данных. Аналитическая обработка в реальном времени, технология обработки информации, включающая составление и динамическую публикацию отчетов и документов. Используется аналитиками для быстрой обработки сложных запросов к базе данных. Служит для бизнес – отчетов по продажам, маркетингу, в целях управления.

Рассматривая данную пирамиду, можно представить себе передачу информации по всем ступеням иерархии системы. Из производственной системы (АСУТП) информация поступает к MES – системам, проходит стадию обработки, а затем уже обработанная информация поступает в ERP – системы, и далее – на уровень высшего менеджмента предприятия (OLAP).

Также уровень АСУТП (автоматизированные системы управления технологическим процессом) можно разделить на два подуровня:

- SCADA (диспетчерское управление)
- PLC (программируемые логические контроллеры)

Верхние уровни (ERP и MES) образуют автоматизированную систему управления предприятием – АСУП (рисунок 1.2).

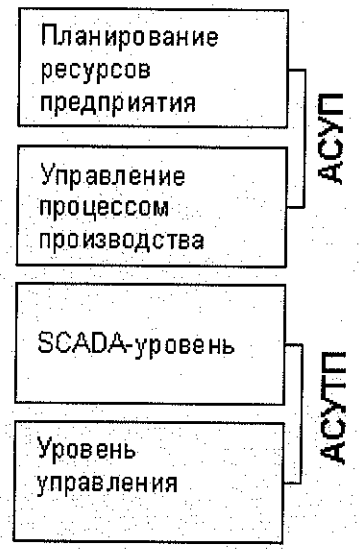


Рисунок 1.2

Также возможно интегрирование APS – системы в информационно-управляющую структуру предприятием.

APS – (Advanced Planning and Scheduling) – система синхронного планирования производства, ориентированная на интеграцию цепи поставок, с учетом всех особенностей и ограничений производства. Идеей и целью системы APS является обеспечение пользователя инструментом, с помощью которого он сможет контролировать и оптимизировать бизнес-процессы.

Данная система служит надстройкой ERP - системы , которая как заменяет, так и расширяет их функциональность в части планирования. APS – система пользуется информацией, содержащейся в транзакционной части ERP. Взаимодействует с MES – системой, которая выступает в качестве источника информации о состоянии запущенных производственных заказов и мощностей. По завершению процесса планирования передает результаты в ERP – систему.

1.2 Понятие MES – системы

Организация ISA определила стандарты, определяющие структуру MES – приложений и их интеграцию в IT – архитектуру компании, независимо от поставщика MES – системы. Стандарт ISA S95 “Enterprise – Control System Integration” определяет уровни модели, описывающей взаимодействие между ERP, MES и уровнем автоматизации производства. Стандарт ISA S88 “General and Site Recipe Models and Representation” определяет модели для batch (рецептурных) задач в таких отраслях промышленности, как пищевая, фармацевтическая, химическая.

Также существует c-MES система (Collaborative Manufacturing Execution System) – одна из моделей системы оперативного управления производством, описывающая производственные системы, принципы их организации и функциональность при взаимодействии с цепочками поставок. Она имеет

чрезвычайно важное практическое значение, потому что построена с учетом особенностей развития индустрии, смещения акцентов от ERP к SCM, а также новых производственных задач в современной глобальной экономике (рисунок 1.3).

Рисунок 1.3

Созданная в 2004 году консорциумом MESA на основе MESA-11. Базовая модель MESA-11 была упрощена: из нее убраны функции, относящиеся к системам ERP и SCM (такие как средства документооборота, инструменты управления техническим обслуживанием и ремонтом оборудования, средства производственного планирования), а взаимодействие производственных информационных систем переориентировано на системы управления цепочками поставок. С точки зрения практического применения данная модель в первую очередь интересна специалистам, принимающим решения о внедрении систем класса ERP, MES, SCM. Она позволяет более точно сформулировать те функции, которые должны быть реализованы на каждом уровне управления.

Исходя из тех изменений, которые произошли в c-MES системе можно увидеть следующее:

- Средства документооборота были выдвинуты в отдельную категорию и теперь редко реализуются в MES – системах
- Инструменты управления техническим обслуживанием и ремонтом оборудования также либо реализуются в виде отдельных систем класса EAM, либо являются модулями в составе ERP
- Информационный обмен все больше концентрируется в области задач управления цепочками поставок и все меньше – в области взаимодействия высокоуровневых и низкоуровневых ИС

• Средства производственного планирования переросли в самостоятельный класс Advanced Planning & Scheduling (APS)

В результате базовая модель MESA -11 была упрощена: из нее были убраны функции, которые относились к системам ERP и SCM, а взаимодействие производственных ИС было переориентировано на системы управления цепочками поставок.

По данным статистики c-MES обеспечивает:

- снижение продолжительности цикла производства в среднем на 45%
- сокращение времени ввода данных на 75%
- количества незавершенной продукции на 24%
- снижение бумажной отчетности между сменами в среднем на 61%
- сокращение времени производственного цикла в среднем на 27%
- объема брака в среднем на 18%
- лишней бумажной документации на 56%

Положения работы MES – систем включают в себя:

- Активация производственных мощностей
- Отслеживание производственных мощностей
- Сбор информации, связанной с производством, от:
    - Систем автоматизации, производственного процесса
    - Сенсоров
    - Персонала
    - Программных систем
- Отслеживание и контроль параметров качества
- Обеспечение персонала и оборудования информацией, необходимой для начала процесса производства
- Установление связей между персоналом и оборудованием в рамках производства
- Установление связей между производством и поставщиками, потребителями, инженерным отделом, отделом продаж и менеджментом
- Реагирование на:
    - Требования по номенклатуре производства
    - Изменение компонентов, сырья и полуфабрикатов, применяемых в процессе производства
    - Изменение спецификации продуктов
    - Доступность персонала и производственных мощностей
- Гарантирование соответствия применимым юридическим актам, например Food and Drug Administration (FDA) США.

Функции c-MES – систем:

1. Сбор и хранение данных – взаимодействие информационных подсистем в целях получения, накопления и передачи технологических данных, циркулирующих в производственной среде предприятия.

2. Управление производственным персоналом – обеспечение возможности управления персоналом.

3. Управление качеством продукции – анализ данных измерений качества продукции в режиме реального времени на основе информации поступающей с производственного уровня

4. Управление производственными процессами – мониторинг производственных процессов, автоматическая корректировка, либо диалоговая поддержка оператора

5. Отслеживание и генеалогия продукции – визуализация информации о месте и времени выполнения работ по каждому изделию. Информация может включать отчеты: об исполнителях, технологических маршрутах, комплектующих, материалах, произведенных переделках, текущих условиях производства

6. Анализ эффективности – предоставление подробных отчетов о реальных результатах производственных операций. Сравнение плановых и фактических показателей.

7. Дентальное планирование – расчет производственных расписаний, основанных на приоритетах, атрибутах, характеристиках и способах, связанных со спецификой изделий и технологией производства

8. Контроль состояния и распределения ресурсов – управление ресурсами производства: технологическим оборудованием, материалами, персоналом, инструментами, методиками работ

9. Диспетчеризация производства – управление потоком производства по операциям, заказам, партиям, сериям, посредством рабочих нарядов

1.3 Основные системы управление производством

Целью каждого предприятия является процветание. Чтобы производство было эффективным – необходимо учитывать все внешние факторы. Одна из проблем множества предприятий – управление производством. Такими факторами являются:

- Недостаток или преизбыток запасов
- Отход от стандартов работы
- Нерациональное использование мощностей и сырья

Все эти факторы приводят к тому, что появляется большой процент брака, огромные временные затраты, а также возможности планирования и контроля получения результата. Это и является главной причиной создания систем управления производством.

В России системы управление производством – пока относительно новое слово в автоматизации. Принципу трехслойной пирамиды соответствует отечественное MES – решение «Фобос», которое занимается планированием и внутрицеховым управлением, обмениваясь данными с ERP – системой. MES – система «Фобос» используется в крупном машиностроении, как правило, в паре с «тяжелыми» ERP – системами – BAAN или SAP. Разработчики системы работают над возможностью интеграции также с «1С:Предприятие» (таблица 1).

Другая система – YSB:Enterprise – предназначена для предприятий СМБ, которыми несколько «не по средствам» приобретать тяжелые ERP – системы. YSB:Enterprise работает по принципу двухслойной пирамиды, где MES – система берет на себя функции и верхнего слоя управления.

Таблица 1

Название системы	Компания - разработчик
«Фобос» (для управления производством дискретного типа, для крупнейших машиностроительных предприятий)	«РТСофт»
Factelligence (APS – система, дискретное производство)	«Весть»
Global MRP/MES	GlobalSystem
Infor: MES Infor:APS	«ЭпикРус»
Ortems (APS – система)	«АНД Проджект»
PolyPlan (гибкое производство, интегрированное производство)	PolyPlan
Preactor (APS – система планирован2ия производственных процессов для СМБ)	«РТСофт»
SyteLine	«Фронтстеп»
T-FACTORY MES	AdAstra
YSB:Enterprise (управление производством дискретного типа, для СМБ)	YSB:Enterprise

Представленная выше таблица дает понятие о том, что на российском рынке одним из действенных вариантов является создание на предприятии системы производственного планирования класса APS или, при достаточно высоком уровне автоматизации производства, MES – системы, обеспечивающие более точное планирование хода производства.

На сегодняшний день существует множество программных продуктов, позволяющих решать те или иные задачи из этого перечня. Например, первые два этапа планирования выполняются, как правило, системами класса ERP, этапы 3 и

4 реализуются средствами MES. Хотя для второго и третьего этапов календарного планирования производства также успешно используют APS.

Выводы по разделу

В данном разделе были рассмотрены теоретические основы необходимые для данной работы. А именно: информационно-управляющая структура производственного предприятия; каждый уровень данной системы; подробно рассмотрена MES – система; приведены примеры других систем управления производством.

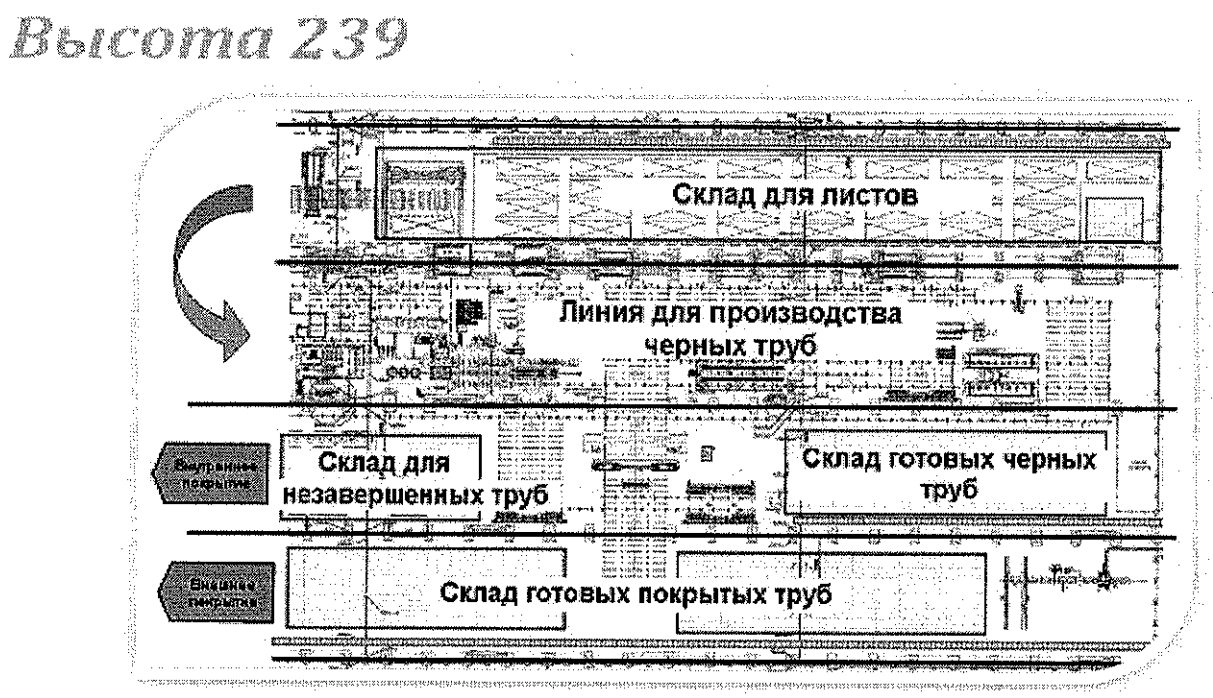
2. ПОСТАНОВКА ЗАДАЧИ

2.1 Структура складского хозяйства ТЭСЦ «Высота 239»

Цех «Высота 239 » - цех по производству сварных труб большого диаметра(508 – 1422 мм) с толщиной стенки до 45 мм на стане ТЭСА.

Рассматривая планировку цеха «Высота 239» (рисунок 2.1) , можно условно разделить производство больших труб на две части:

- Работа со стальным листом, из которого делаются трубы
- Операции с готовой трубой: шлифовка, сварка, проверка ультразвуком рентгеном, мытье, итоговая формовка и прочее



Структура складского хозяйства ТЭСЦ «Высота 239»

Рисунок 2.1

Посреди цеха расположен диспетчерский центр, где находится главный пульт управления производством, все этапы которого отображаются на огромном мониторе. Основное оборудование цеха – это два огромных пресса, которые гнут 2 – х сантиметровый стальной лист в почти замкнутую окружность, которая в дальнейшем станет готовой трубой. Для идеального сгиба по окружности применяется газовая резка стальных полос, которые привариваются к листу.

Также имеются несколько складов для разного вида продукции:

- Склад для листов
- Склад для незавершенных труб
- Склад готовых черных труб
- Склад готовых покрытых труб

В системе WTMS работают 11 кранов (+ 5 технологических) выполняющих:
- Разгрузку/погрузку вагонов
- Задачу материалов в линии
- Перемещение материалов

Рабочими кранами являются краны под номерами: 1, 2, 3, 4, 5, 6, 7, 8, 9, 15, 16. В терминале должна происходить только с данными кранами.

2.2 Взаимодействие с ERP – системой

MES – системы и АСУ ТП получают информацию от PLC и направляют ее как в исходном, так и интегрированном виде в базы данных реального времени. Обработка накопленных в них данных осуществляется в реальном масштабе времени протекания процессов управления производством, и серьезные временные задержки при этом допустимы: время обработки данных не должно существенно увеличиваться с ростом ее объема.

Данные от PLC о состоянии оборудования и технологических процессов направляются в базы данных SCADA – систем и затем, после процедур фильтрации, интегрируется с базами данных MES. На уровне MES информация также фильтруется и обобщается, при этом возможна генерация в реальном масштабе времени некоторых технико-экономических показателей деятельности предприятия. По запросу от ERP – системы информация из баз данных MES поступает в их базы данных. Обратно же, информация поступает от ERP к MES и дал2ее на уровень PLC. Происходит сложный процесс обмена информацией между базами данных различных уровней системы управления предприятием. Характеристики этого процесса оказывают существенное влияние на качество процессов управления.

С уровня ERP на уровень MES поступает план и нормативно-справочная информация, на основе которых в DPS, MES – системы генерируются конкретные планы загрузки станков и рабочей силы – сменные расписания.

С уровня MES на уровень ERP направляется информация о количестве произведенной продукции, проценте брака, диаграммы состояния и загрузки оборудования , данные о состоянии цеховых запасов и рабочей силы.

Взаимодействие уровней MES и ERP возможно на основе программируемого интервала времени и на основе ожидаемых событий и изменения данных. Эти методы взаимодействия достаточно гибко программируются в процессе настройки систем.

Из ERP – системы передаются:

- Заказы
- Материалы
- Вагоны
- Информация по допуску сотрудников (сертификации)

2.3 Структура и описание модуля

Пользовательские требования

Системные диалоги разработаны для использования в среде MS Windows соответствуют стандарту DIN EN ISO 9241 – 110 Ergonomics of Human System Interaction, т.е. основаны на стандартных элементах диалога, таких как поля ввода, выпадающие списки, кнопки и т.д.

Требуемое разрешение экрана – 1280 x 1024 пикселей и по меньшей мере 32768 (15 bit) цветов.

Описание

Этот модуль для крановщика, предназначен для транспортировки листов и труб. Главная часть модуля логически разделена на четыре части (рисунок 2.4):

- Информационная панель
- Панель сообщений
- Панель рабочих заданий
- Панель графического отображения/обзора склада

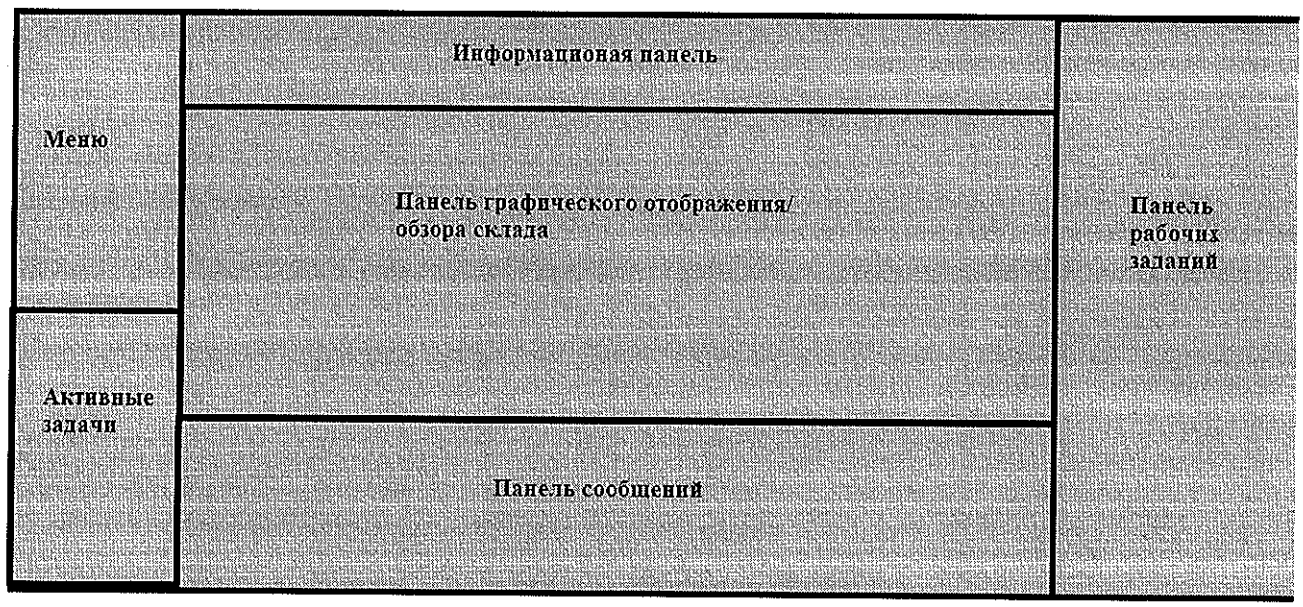


Рисунок 2.4

Информационная панель показывает следующую информацию по крану:

- Номер крана
- Штабель

- Режим работы
- Вес
- Расчетный вес (если кран взял несколько листов или труб)
- Текущая логическая и физическая позиция

Также на информационной панели происходит переключение между кранами.

Внизу модуля имеются кнопки для крановщика. В зависимости от рабочего режима крана (и других условий) некоторые кнопки могут быть неактивны.

- Поднять
- Опустить
- Потеря
- Вернуть
- Смена обзора
- АвтоОбзор
- Показать РЗ(рабочее задание)

На панели графического отображения/обзора склада показано графическое представление траверсы и то, что на ней сейчас находится. Также есть возможность обзора склада. Графический режим и обзор склада представлены в одном окне и соответственно есть возможность переключения между вкладками.

В обзоре склада отображается план всего цеха. Также путь, по которому кран должен пройти, чтобы выполнить рабочее задание.

Панель сообщений содержит:

- Тип сообщения
- Число, когда сообщение пришло
- Дата, когда сообщение пришло
- Само сообщение

Панель рабочих заданий должна содержать таблицу, состоящую их пяти столбцов:

- Количество материала
- Номер материала
- Источник (откуда взять материал)
- Цель (куда доставить материал)
- Вес, кг

Также на панели должны содержаться кнопки:

- Взять РЗ
- Отмена выполнения РЗ

Выводы по разделу

В конце главы подведем итог и выделим самые главные моменты постановки задачи. Модуль разбит на несколько блоков. Каждый из блоков выполняет

определенную задачу. Информация по материалам и транспортным средствам передается из ERP –системы.

3. РЕАЛИЗАЦИЯ МОДУЛЯ

3.1 Интерфейс

Для решения данной задачи использовался программа Microsoft Visual Studio 2012. Интерфейс создавался с помощью Windows Form и панели ее элементов. Интерфейс представлен на (рисунок 3.1).

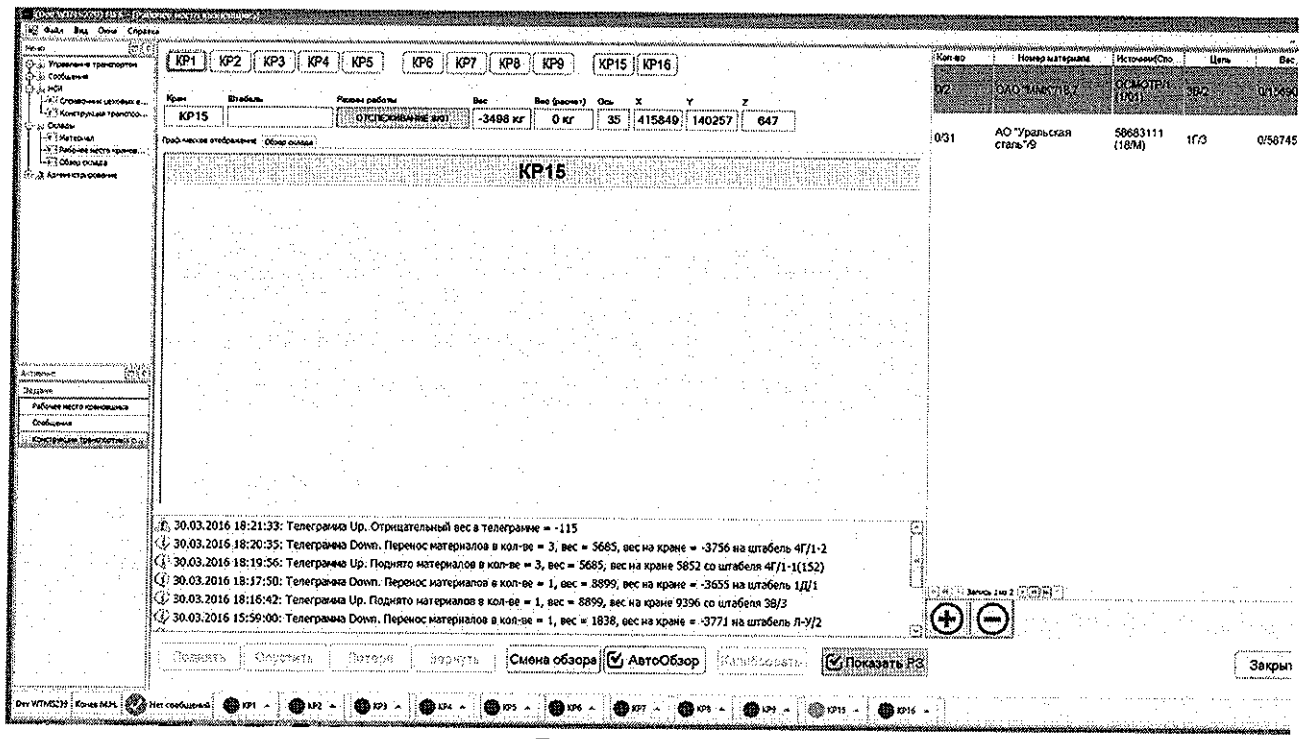


Рисунок 3.1

Главная панель рабочего места крановщика состоит из двух вкладок:

- Графическое отображение
- Обзор склада

На вкладке «Графическое отображение» отображается штабель, над которым сейчас находится кран. Причем, в «Графическом отображении» отображаются 7 верхних материалов штабеля (рисунок 3.2).

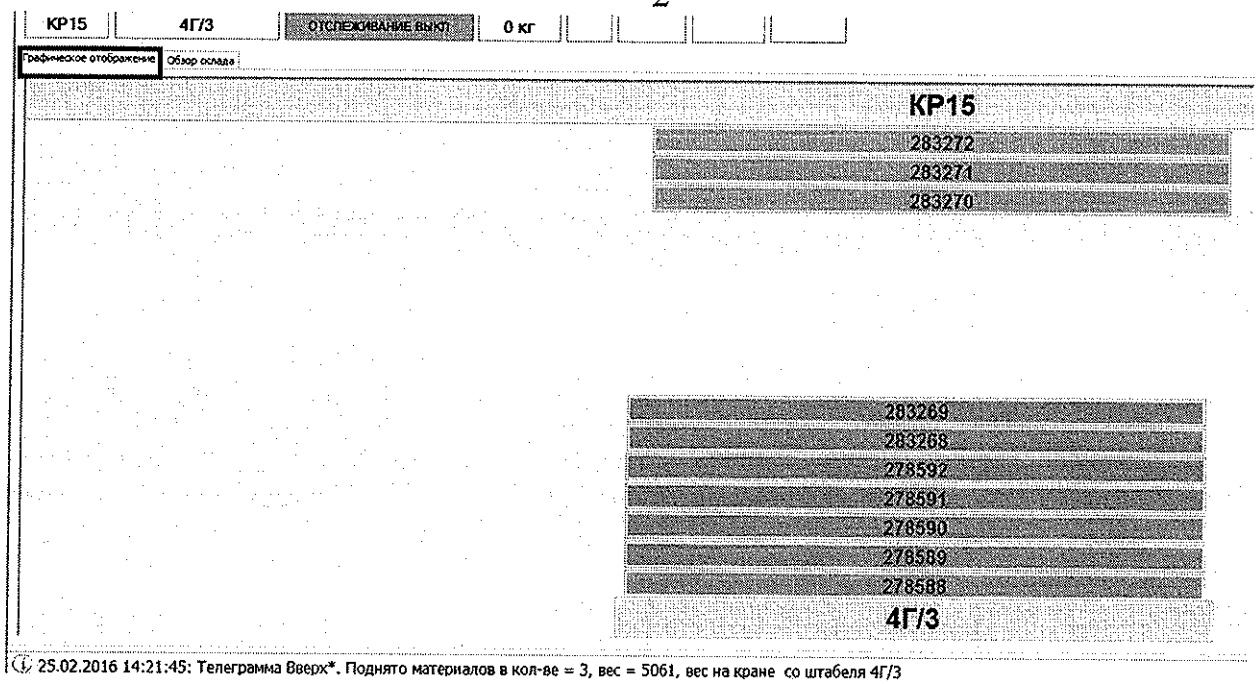


Рисунок 3.2

На вкладке «Обзор склада» отображается схема пролета, в котором находится кран. Цвета штабелей указываются на степень заполненности штабеля материалами (рисунок 3.3).

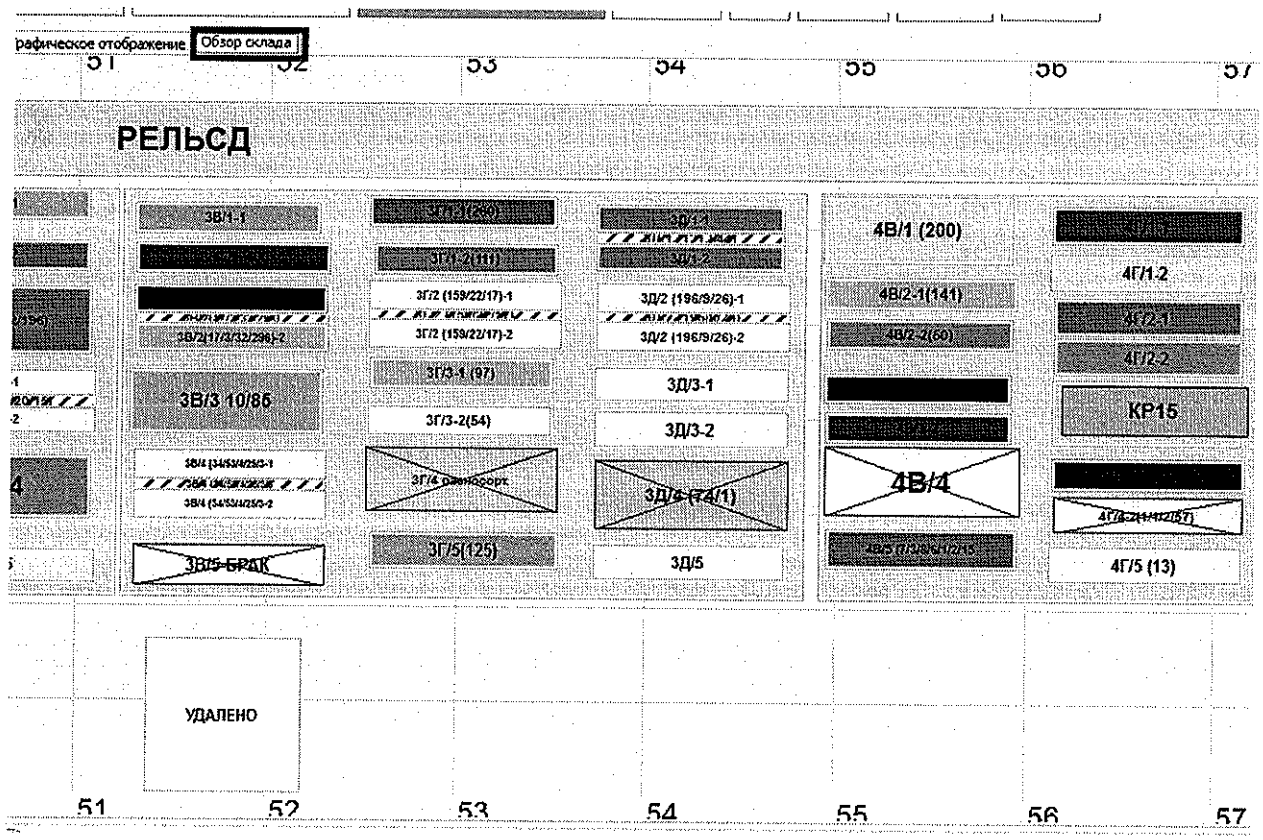


Рисунок 3.3

Обзор склада является отдельным модулем и на рабочем месте крановщика он представлен, как тот же самый модуль, только в ограниченной функциональности.

3.2 Информационная панель с параметрами крана

В верхней части экрана рабочего места крановщика расположена информационная панель, на которой отображается имя крана, наименование штабеля, над которым он сейчас находится, режим работы крана, вес поднятый краном, а также координаты крана в пролете (рисунок 3.4).

Рисунок 3.4

Режим работы крана бывают двух видов:

- Отслеживание ВЫКЛ
- Отслеживание ВКЛ

Если режим работы «Отслеживание ВЫКЛ», то перемещение материалов не фиксируется системой, работа идет вчерную.

Если режим крана «Отслеживание ВКЛ», то перемещение материалов отслеживается (рисунок 3.5).

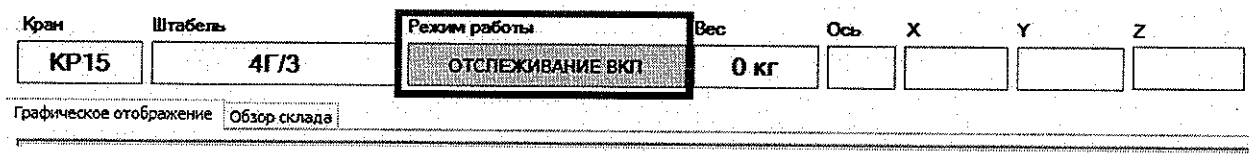


Рисунок 3.5

3.3 Таблица рабочих заданий

Если перемещение материалов выполняется по рабочим заданиям, созданным бригадиром, то для этого необходимо взять задание в работу. Для этого на рабочем месте крановщика есть таблица с рабочими заданиями. По умолчанию эта таблица скрыта, чтобы ее увидеть или скрыть, необходимо нажать кнопку «оказать РЗ» (рисунок 3.6).

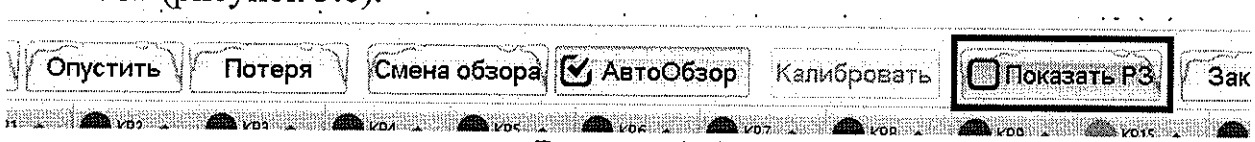


Рисунок 3.6

Если в таблице появятся новые рабочие задания, а она будет находиться в свернутом виде, кнопка изменит свой цвет.

В таблице рабочих заданий возможно как взять рабочее задание в работу , так отменить его выполнение определенным краном (рисунок 3.7).

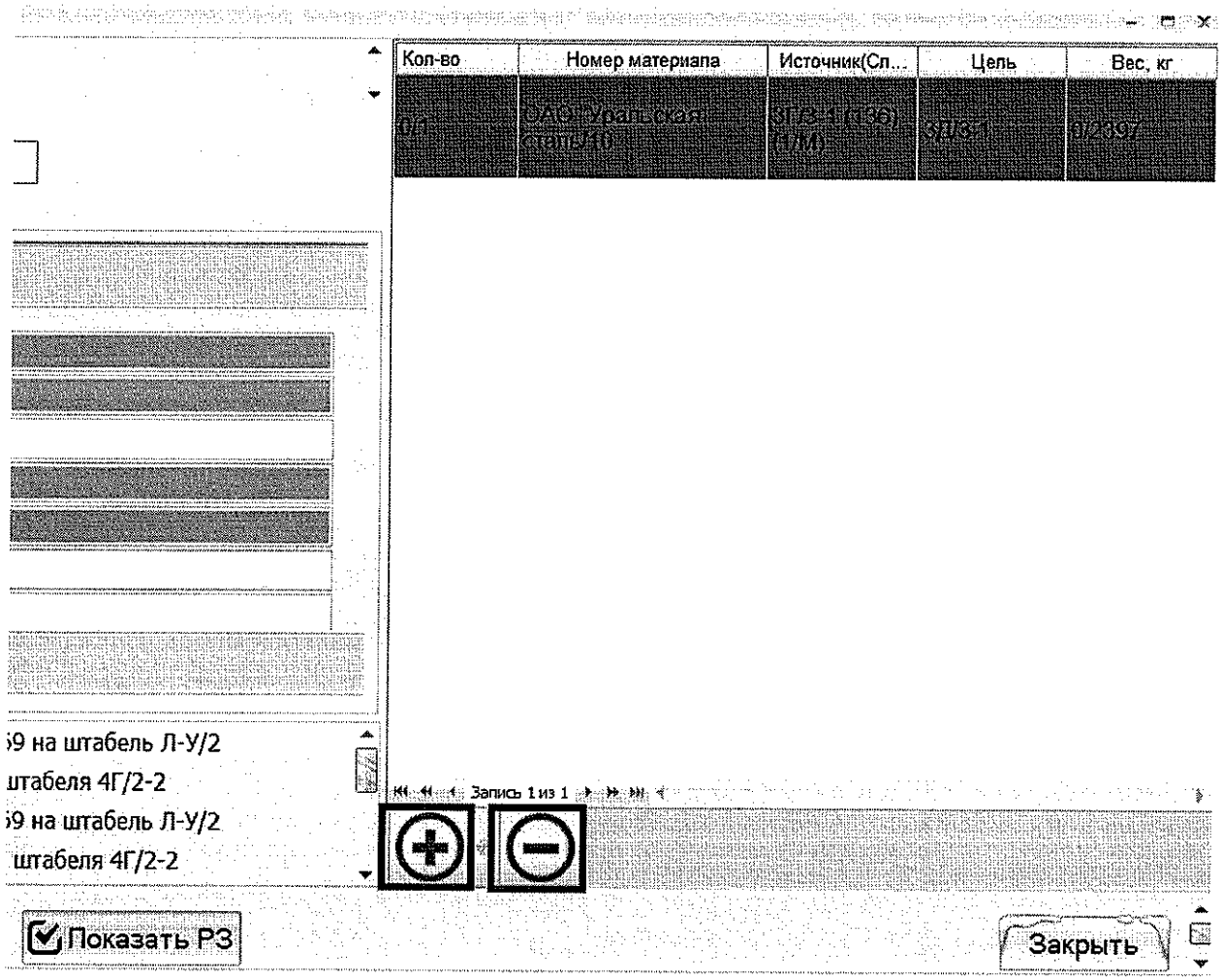


Рисунок 3.7

При взятии задания в работу оно должно поменять свой цвет на оранжевый (рисунок 3.8).

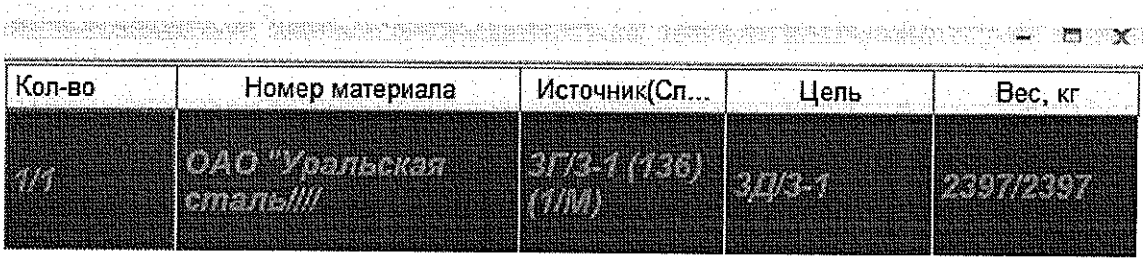


Рисунок 3.8

При возвращении задания должно поменять свой цвет на черный.

Взятие задания в работу и его возвращение сопровождается появлением сообщений в списке сообщений на графическом отображении штабеля (рисунок 3.9).

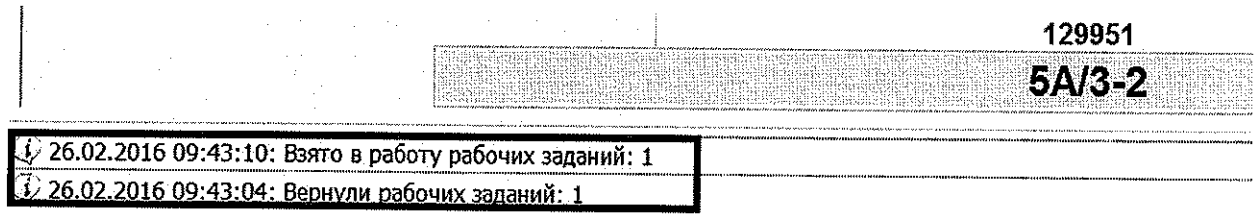


Рисунок 3.9

При перемещении материала по рабочим заданиям на вкладке «Обзор склада» появляются подсказки для движения крана: бирюзовый маркер обозначает место, где лежит материал, который нужно переместить, красный маркер – место куда необходимо перенести материал. Также от крана, взявшего задание в работу, до текущего места назначения проводится стрелка для указания направления движения (рисунок 3.10).

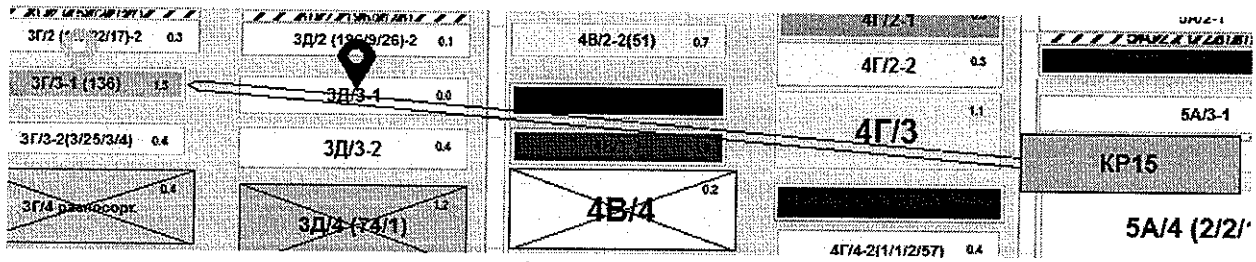


Рисунок 3.10

Также возможна работа без рабочих заданий. Если в режиме работы крана с отслеживанием выполняются перемещения материалов без рабочего задания, то эти перемещения все равно отслеживаются, так как рабочие задания создаются в системе за кадром по данным, которые отправляются при поднятии и опускании материала. Эти задания не появляются в таблице рабочих заданий, но фиксируются в системе, по ним также выводятся сообщения.

3.4 Сообщения

Каждое событие, связанное с перемещением материала, отображается в списке сообщений, который располагается на вкладке «Графическое отображение» под отображением штабеля. Эти сообщения появляются при перемещении материалов краном по телеграммам, которые он отправляет, а также при корректировке материалов на кране с помощью кнопок «Поднять», «Отпустить» и «Потеря» (рисунок 3.11).

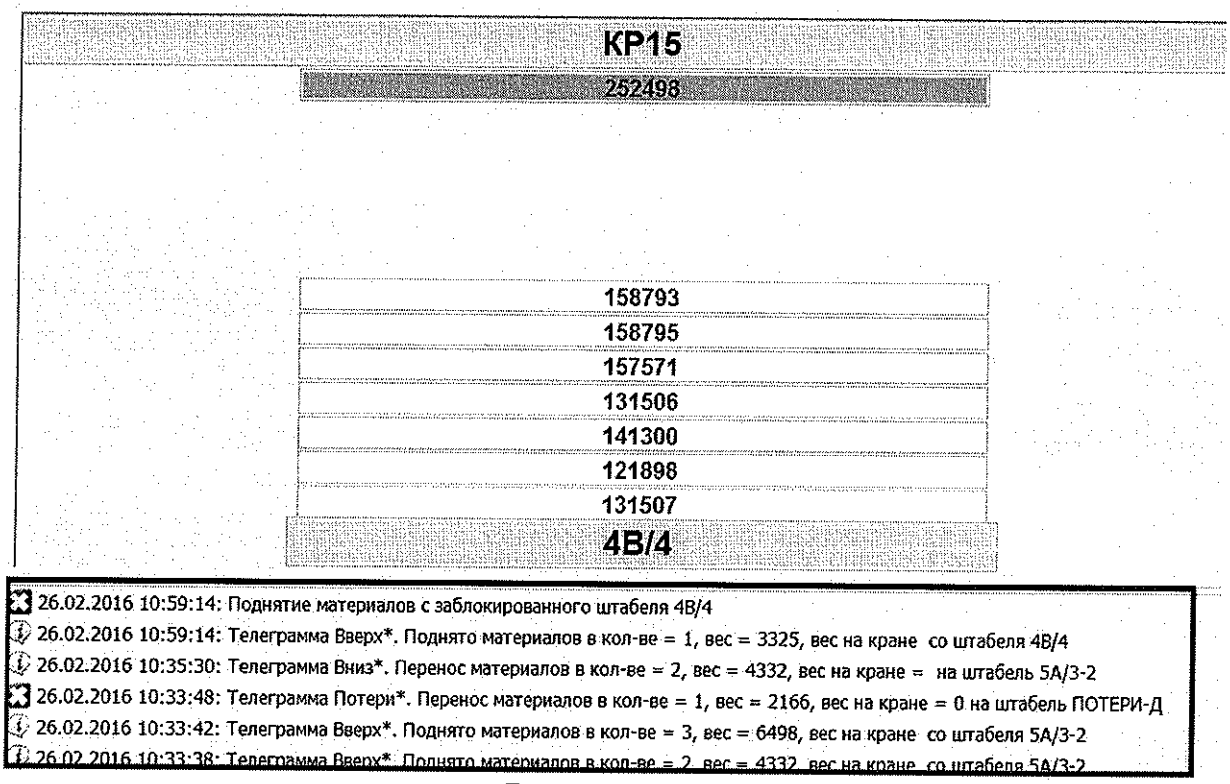


Рисунок 3.11

Сообщения бывают информационные и обозначаются значком и критические .

При появлении критического сообщения в списке, оно будет мигать красным.

3.5 Реализация системы

При проектировании системы, модуль был разбит на 11 классов (рисунок 3.12).

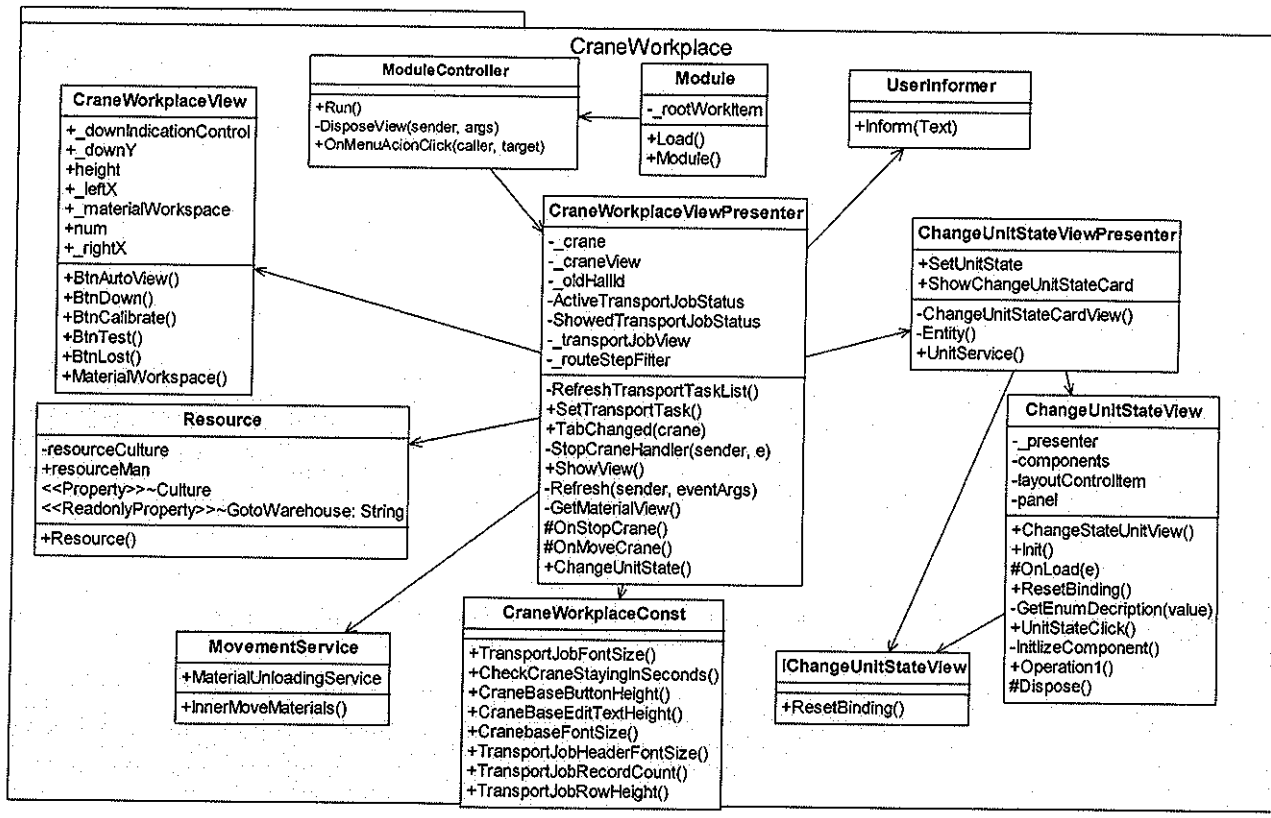


Рисунок 3.12

3.5.1 Class UserInformer

Этот класс выводит основную информацию крановщику (рисунок 3.13).

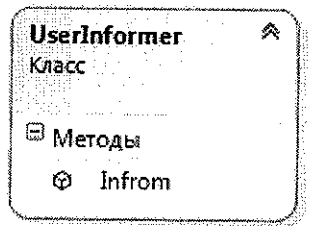


Рисунок 3.13

Методы класса:

1. Inform – метод получающий информацию и выводящий ее на экран

3.5.2 Class ModuleController

Данный класс предназначен для запуска модуля (рисунок 3.14).

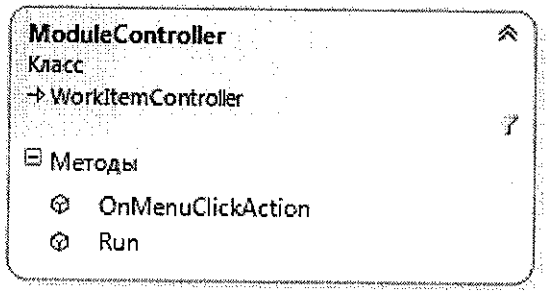


Рисунок 3.14

Методы класса:

1. OnMenuClickAction – метод определяющий, какой модуль нужно запускать и передающий информацию и параметры в метод Run.
2. Run – метод, с помощь которого происходит инициализация модуля и его запуск с определенными параметрами.

3.5.3 Class MovementService

Этот класс отвечает за движение кранов и материалов по цеху (рисунок 3.15).

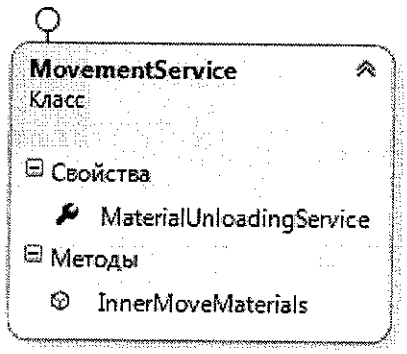


Рисунок 3.15

Свойства класса:
1. MaterialUnloadingService – метод, отслеживающий отгрузку материала.

Методы класса:
1. InnerMoveMaterials - метод, отслеживающий правильность внутреннего перемещения материалов.

3.5.4 Class Module

Этот класс предназначен для запуска модуля (рисунок 3.16).

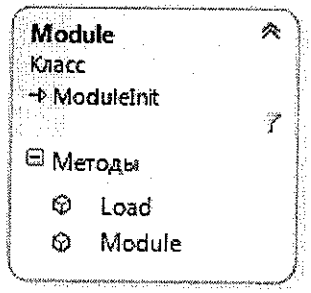


Рисунок 3.16

Методы класса:

1. Load – метод класса, отвечающий за загрузку компонентов модуля.
2. Module – метод класса, отвечающий за то, какие компоненты модуля надо загрузить.

3.5.5 Class CraneWorkplaceViewPresenter

Данный класс является основным и отвечает за управление модулем (рисунок 3.17).

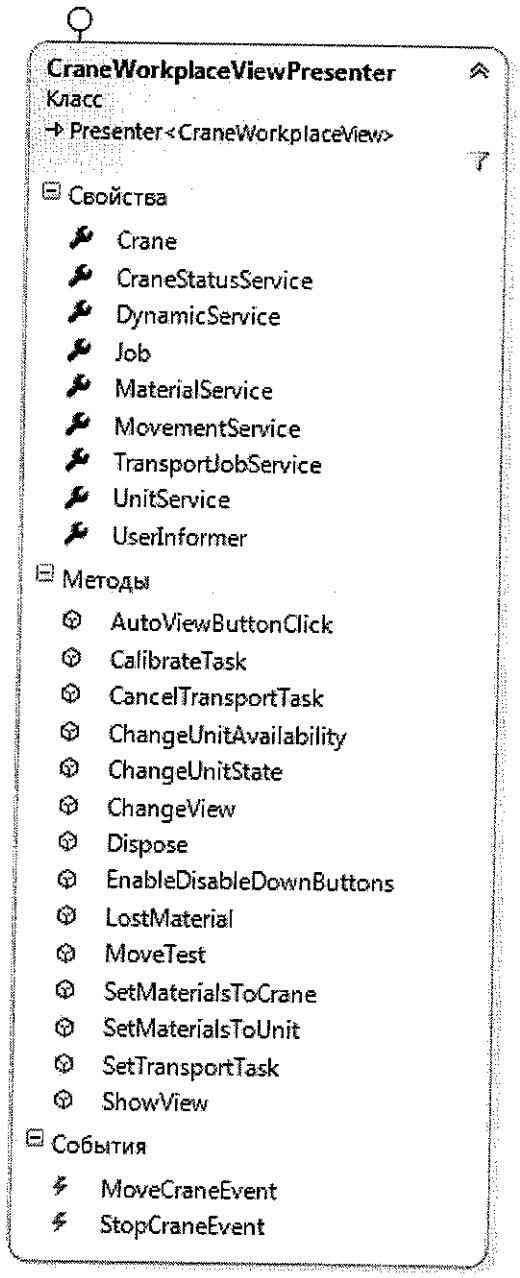


Рисунок 3.17

Свойства класса:

1. Crane – свойство класса, предоставляющее общий доступ к рабочим кранам.

2. CraneStatusService – свойство класса, предоставляющее информацию о состоянии кранов.

3. DynamicService – свойство класса, предоставляющее информации о кране при его перемещении.

4. Job – свойство класса, получающие доступ к рабочим заданиям.

5. MaterialService – свойство класса, получающее общий доступ к материалам на складе.

6. MovementService – свойство класса, получающее доступ к движениям крана.

7. TransportJobService – свойство класса, предоставляющее доступ к рабочим заданиям.

8. UnitService – свойство класса, предоставляющее доступ к обслуживанию оборудования.

9. UserInformer – свойство класса, предоставляющее доступ к классу UserInformer для вывода информации на экран.

Методы класса:

1. ShowView – функция отвечающая за открытие определенного окна.
2. SetTransportTask – функция отвечающая за транспортные задания.
3. SetMaterialsToUnit – функция, отвечающая за определенный материал.
4. SetMaterialsToCrane – функция, отвечающая за определенный кран.
5. MoveTest – функция отвечающая за передвижение крана.
6. LostMaterial – функция, отвечающая за потерю материала.
7. EnableDisableDowtButtons – функция отвечающая за то, какая активность кнопок.
8. Dispose – функция, отвечающая за расположение, выбранного предмета.
9. ChangeView – функция отвечающая за смену вида.
10. ChangeUnitState – функция, отвечающая за изменение состояния блока на интерфейсе модуля.
11. ChangeUnitAvailability – функция, отвечающая за изменение единицы измерения.
12. CancelTransportTask – функция, вызывающаяся при закрытии транспортного задания.
13. CalibrateTask – функция, вызывающаяся при нажатии кнопки «Калибровка».
14. AutoViewButtonClick – функция, вызывающаяся при нажатии кнопки «Автоматический просмотр».

3.5.6 Class ChangeStateUnitView

Данный класс отвечает за интерфейс при управлении определенными блоками (рисунок 3.18).

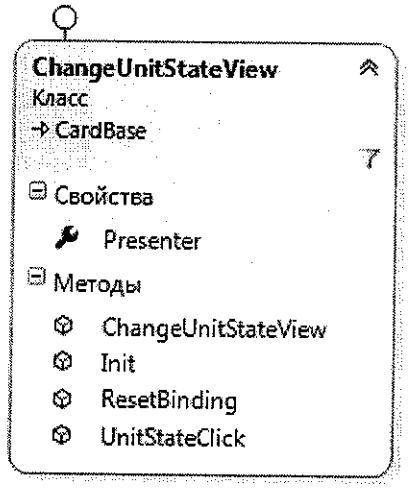


Рисунок 3.18

Свойства класса:

1. Presenter – свойство, получающее доступ к ChangeUnitStateViewPresenter.

Методы класса:

1. ChangeUnitStateView – метод отвечающий, обращающийся сам к себе при повторном использовании класса.
2. Init – метод класса, отвечающий за инициализации блоков в интерфейсе модуля.
3. ResetBinding – метод , отвечающий за сброс привязки интерфейса.
4. UnitStateClick - метод отвечающий за то, к какому из блоков в интерфейсе модуля, обратился пользователь.

3.5.7 Class ChangeUnitStateViewPresenter

Данный класс отвечает за управление блоками модуля (рисунок 3.19).

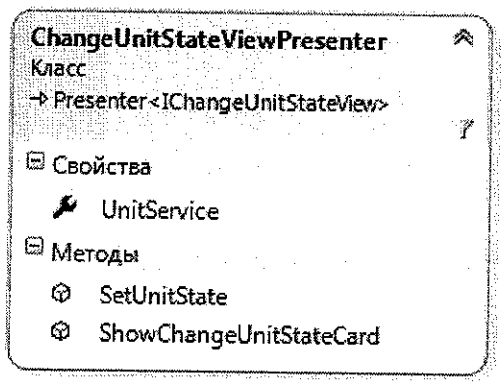


Рисунок 3.19

Свойства класса:

1. UnitService – структура, хранящая информацию об обслуживании кранов.

Методы класса:

1. SetUnitState – метод класса, отвечающий за определенный блок в интерфейсе модуля.

2. ShowChangeUnitStateCard – метод класса, отвечающий за общее состояние блоков в интерфейсе модуля.

3.5.8 Class CraneWorlplaceView

Данный класс отвечает за интерфейс модуля (рисунок 3.20).

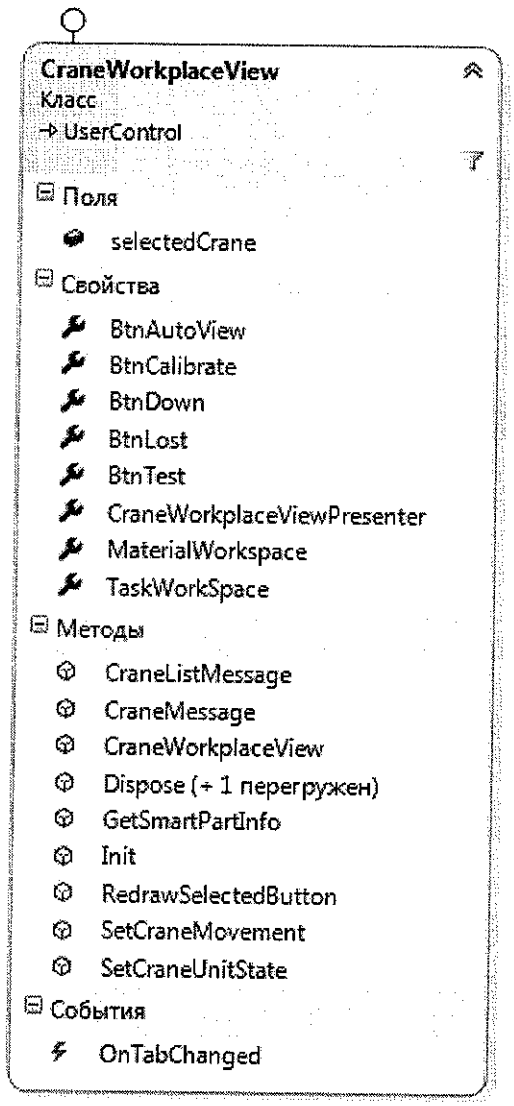


Рисунок 3.20

Поля класса:

1. selectedCrane – переменная, содержащая, какой из кранов был выбран для просмотра его состояния или для работы.

Методы класса:

1. CraneListMessage – метод класса, отвечающий за посылку информации каждым краном о своих передвижениях и действиях.

2. CraneMessage – метод класса, отвечающий за посылку информации о своих передвижениях и действиях в CraneListMessage.

3. CraneWorkplaceView – метод класса, обращающийся сам к себе при повторном использовании класса.

4. GetSmartPartInfo – метод класса, отвечающий за получение определенной информации.

5. Init – метод класса, отвечающий за инициализацию класса CraneWorkplaceView.

6. RedrawSelectedButton – метод класса, отвечающий за перерисовку выбранной кнопки.

7. SetCraneMovement – метод класса, отвечающий за движение крана.

8. SetCraneUnitState – метод класса, отвечающий за состояния кранов.

Свойства класса:

1. BtnAutoView – свойство, происходящее при нажатии кнопки «Автоматический просмотр».

2. BtnCalibrate – свойство, происходящее при нажатии кнопки «Калибровка».

3. BtnDown – свойство, происходящее при нажатии кнопки «Назад».

4. BtnLost – свойство, происходящее при блокировке какой либо из кнопок.

5. BtnTest – свойство, происходящее при нажатии кнопки «Тестирование».

6. CraneWorkplaceViewPresenter – свойство получающее доступ к

7. MaterialWorkspace – свойство, отвечающее за материал рабочей области.

8. TaskWorkSpace – свойство, отвечающее за задания рабочей области.

3.5.9 Class IChangeUnitStateView

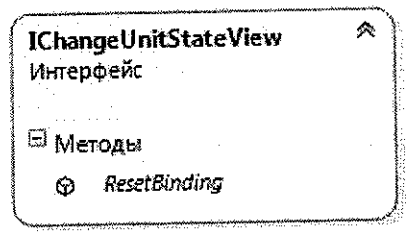


Рисунок 3.21

Методы класса:

1. ResetBinding – метод , отвечающий за сброс привязки интерфейса.

3.5.10 Class CraneWorkplaceConst

В данном классе определены основные константы (рисунок 3.21).

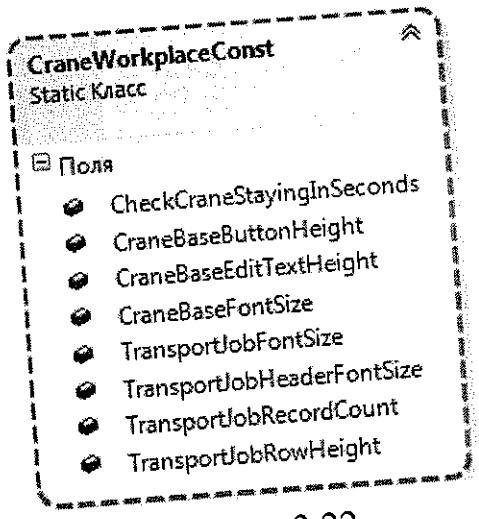


Рисунок 3.22

Поля класса:

1. CheckCraneStayingInSeconds – хранится переменная, в которой обозначается через сколько секунд кран считается остановившимся.
2. CraneBaseButtonHeight – хранится переменная, в которой обозначается высота базовой кнопки крана.
3. CraneBaseEditTextHeight - хранится переменная, в которой обозначается размер шрифта при изменении высоты в крановом терминале.
4. CraneBaseFontSize – хранится переменная, в которой обозначается размер шрифта для кранового терминала.
5. TransportJobFontSize – хранится переменная, в которой обозначается размер шрифта в транспортных заданиях.
6. TransportJobHeaderFontSize – хранится переменная, в которой обозначается размер шрифта заголовка в транспортных заданиях.
7. TransportJobRecordCount – хранится переменная, в которой обозначается количество строк в транспортных заданиях.
8. TransportJobRowHeight – хранится переменная, в которой обозначается высота строки транспортного заданиях.

3.6 Тестирование модуля

Было проведено тестирование как модуля отдельно, так и в общей системе управления производством.

Более подробно с результатами тестирования можно ознакомиться в Приложении 1 «Карта тестирования рабочего места крановщика».

Выводы по разделу

В результате проведенной в данной главе работе спроектирован и разработан модуль «Рабочее место крановщика». Интегрирован в общую систему управления производством и пущен в производство. Также проведено тестирование модуля, которое показало положительные результаты.

Заключение

В результате выполнения данной работы разработан модуль «Рабочее место крановщика» для ОАО «ЧТПЗ» цех «Высота 239». Проведено тестирование и пробные запуски, которые показали работоспособность системы, как отдельно, так и в общей системе. Модуль успешно интегрирован в общую систему управления производством и в дальнейшем, пущен в производство. Минусом данного модуля является то, что бригадир может сидеть только за стационарным компьютером в отдельном помещении и не может перемещаться по цехе, внимательнее наблюдая за всеми процессами. Для улучшения данной системы можно попытаться сделать сенсорное управление, с помощью которого бригадир мог бы ходить по всему цеху и контролировать производственный процесс.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1 Ekstrand, M. Collaborative Filtering Recommender Systems / M. Ekstrand, J. Riedl, J. Konstan // Foundations and Trends in Human–Computer Interaction . – 2010. №2. –P. 88–114. – Дата обновления: 24.06.2010. URL: http://files.grouplens.org/papers/FnT%20CF%20Recsys%20Survey.pdf (дата обращения: 05.06.2016).

2 Pazzani, M. Content-based Recommendation Systems / M. Pazzani, D. Billsus // – Дата обновления: 04.12.2012. URL: http://www.fxpal.com/publications/FXPAL-PR-06-383.pdf (дата обращения: 05.06.2016).

3 Методы сбора оценок. – Дата обновления: 20.03.2015. URL: https://yandex.ru/blog/company/92883 (дата обращения: 05.06.2016).

4 Belavkin, R. Lecture 11: Feed-Forward Neural Networks /R. Belavkin // – Дата обновления: 22.07.2014. URL: http://www.eis.mdx.ac.uk/staffpages/rvb/teaching/BIS3226/hand11.pdf (дата обращения: 05.06.2016).

5 Метод обратного распространения ошибки. – Дата обновления: 12.03.2011. URL: https://en.wikipedia.org/wiki/Backpropagation (дата обращения: 05.06.2016).

6 Dumoulin, V. A guide to convolution arithmetic for deep learning / V. Dumoulin, F. Visin // – Дата обновления: 29.04.2013. URL: http://arxiv.org/pdf/1603.07285v1.pdf (дата обращения: 05.06.2016).

7 Функции активации. – Дата обновления: 11.01.2010. URL: https://en.wikipedia.org/wiki/Activation_function (дата обращения: 05.06.2016).

8 Функции потерь. – Дата обновления: 18.11.2014. URL: https://en.wikipedia.org/wiki/Loss_functions_for_classification (дата обращения: 05.06.2016).

9 Bottou, L. Stochastic Gradient Descent Tricks / L. Bottou // – Дата обновления: 02.10.2015. URL: http://research.microsoft.com/pubs/192769/tricks-2012.pdf (дата обращения: 05.06.2016).

10 Srivastava, N. Dropout: A Simple Way to Prevent Neural Networks from Overfitting / N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov // Journal of Machine Learning Research. – 2014. №15. –P. 1929-1958.

11 А. Лукин. Введение в цифровую обработку сигналов (математические основы) / Алексей Лукин // – Дата обновления: 05.07.2007. URL: http://audio.rightmark.org/lukin/dspcourse/dspcourse.pdf (дата обращения: 05.06.2016).

12 Lyon, D. The Discrete Fourier Transform, Part 2: Radix 2 FFT / D. Lyon // Journal of object technology. – 2009. №5. – P. 21-3313 Коннолли, Базы данных. Проектирование, реализация и сопровождение. Теория и практика. 3-е издание. / Коннолли, Томас, Бегг, Карелии. – Санкт-Питербург : Пер. с англ. – М. : Издательский дом "Вильяме", 2003. — 1440 с.

14 Шлее, М. Qt 4.8. Профессиональное программирование на C++ / М. Шлее – Санкт-Питербург.: БХВ-Петербург, 2012. — 912 с.

15 Конвей, Р. В. Теория Расписаний / Конвей Р. В., Максвелл В. Л., Миллер Л. В. – Москва, 1975 – 359 с.

16 Лазарев, А. А. Теория расписаний задачи и алгоритмы. / Лазарев А. А. Гафаров Е. Р. —Москва, 2011. – 222 с.

17 Merkle, D. Ant Colony Optimization for Resource-Constrainted Project Scheduling / Merkle D., Middendorf M., Schmeck H // IEEE Transactions on Evolutionary Computation., 2002. Vol. 6. No. 4. P. 333–346.

18 Graham, R. L. Optimization and approximation in deterministic sequencing and scheduling: A survey. Annals of Discrete Mathematics / Graham R. L., Lawler E. L., Lenstra J. K., Rinnooy Kan A. H. G., 1979, том 5, с. 287-326.

19 URL: http://doc.qt.io/.

ПРИЛОЖЕНИЕ 1
Карта тестирования рабочего места крановщика

	Описание теста	Процедура проведения теста	Ожидаемый результат	Дата тестирования	Результат	Примечание
	Открытие Рабочего Места Крановщика	Запустить РМК через ярлык приложения	Открывается окно приложения с РМК, в котором находятся две вкладки - "Графическое отображение" и "Обзор склада" В информационной панели отображается номер крана, положение, режим работы и вес на кране	11.03.2016	Пройден	
	Автоматическое переключение с вкладки "Обзор склада" на "Графическое отображение"	Кнопка "АвтоОбзор" нажата Открыт "Обзор склада" Кран стоит над штабелем в течение времени, заданного в настройках системы	Происходит переключение на вкладку "Графическое отображение", где отображается штабель, над которым кран находится	11.03.2016	Пройден	Время, через которое считается, что кран остановился может быть изменено Администратором в настройках системы
	Автоматическое переключение с вкладки "Графическое отображение" на "Обзор склада"	Кнопка "АвтоОбзор" нажата Открыто "Графическое отображение" Кран начинает	Происходит переключение на вкладку "Обзор склада"	11.03.2016	Пройден	Расстояние, через которое считается, что кран начал движение может быть изменено Администра

	движение				тором
Ручное переключение между вкладками "Графическое отображение" на "Обзор склада"	Кнопка "АвтоОбзор" отжата Нажать кнопку "Смена обзора"	Переключение с одной вкладки на другую	11.03.2016	Пройден	
Поднятие одного материала со штабеля	Находясь на вкладке "Графическое отображение" над непустым штабелем, нажать кнопку "Поднять"	Один верхний лист со штабеля появляется на траверсе На штабеле отображаются оставшиеся верхние материалы В списке сообщений появляется телеграмма "Up" с количеством материалов 1	11.03.2016	Пройден	
Опускание всех материалов крана на штабель	Находясь на вкладке "Графическое отображение", нажать кнопку "Опустить"	Все материалы, находящиеся на траверсе перемещаются на штабель, находящийся под краном В списке сообщений появляется телеграмма "Вниз" с количеством опущенных материалов, штабелем, куда опущены материалы и весом опущенных материалов	11.03.2016	Пройден	
Поднятие трёх материалов	Находясь на вкладке "Графическо	При первом нажатии кнопки поднимается	11.03.2016	Пройден	

	на кран	е отображение " над непустым штабелем, трижды нажать кнопку "Поднять"	первый верхний материал со штабеля. В сообщениях появляется телеграмма "Вверх" с количеством материалов 1, весом одного материала и штабелем, с которого материал был взят. При повторном нажатии поднимается следующий из верхних материалов штабеля. В сообщениях появляется телеграмма "Вверх" с количеством материалов, равным 2, весом, равным весу двух поднятых материалов, и штабелем, откуда они были взяты. При третьем нажатии поднимается слудеющий из верхних материалов штабеля. В сообщениях появляется телеграмма "Вверх" с количеством материалов, равным 3, весом, равным весу трёх поднятых материалов, и штабелем, откуда			

			они были взяты.			
	Потеря материала	Находясь на вкладке "Графическое отображение" нажать кнопку "Потеря"	Нижний лист на кране перемещается на штабель "Потеря" пролёта, в котором находится кран В сообщениях появляется телеграмма "Потери" с количеством материалов, равным одному, весом, равным весу потерянного материала и со штабелем, куда перемещён потерянный материал. Это сообщение подкрашивается красным цветом в течение нескольких секунд после появления.	11.03.2016	Пройден	
	Возврат одного материала на штабель	Находясь на вкладке "Графическое отображение", нажать кнопку "Вернуть"	Нижний лист на кране перемещается на штабель, над которым находится кран В сообщениях появляется телеграмма "Back" с количеством материалов, равным 1, весом, равным весу возвращённого материала, и со штабелем, куда перемещён материал. Это	11.03.2016	Пройден	

			сообщение подкрашивается красным цветом в течение нескольких секунд после появления.			
	Открытие РЗ	Нажать на кнопку "Показать РЗ"	Открывается Грид РЗ, в котором отображаюся задания того пролёта, в котором находится кран	11.03.2016	Пройден	
1	Взятие РЗ в работу	В открытом гриде РЗ выбрать строку с нужным рабочим заданием Нажать кнопку "Взять в работу"	Задание, взятое в работу становится оранжевым На обзоре склада появляются указатели Источника и Цели рабочего задания От крана до штабеля-источника рисуется стрелка В сообщениях появляется информация о взятии задания в работу	11.03.2016	Пройден	
2	Перемещение материала по заданию	Переместить кран к штабелю-источнику РЗ Выполнить поднятие материалов, подсвеченных зелёным Переместить кран к штабелю цели Нажать "Отпустить"	После поднятия материалов со штабеля-источника стрелка на обзоре склада рисуется от крана до штабеля-цели После опускания материала на штабель цель задания на перещённые материалы исчезают из грида РЗ На обзоре склада	11.03.2016	Пройден получилось только на локалхосте	

			исчезают маркеры, подсказывающие движение крану			
2	Перемещение материала, на которое есть задание, без взятия его в работу	Создать задание на перемещение материала из одного штабеля в другой Без взятия в работу этого задания поднять материал с заданием со штабеля-источника	Задание на этот материал автоматически возьмётся в работу краном	11.03.2016	Пройден получилось только на локалхосте	
1	Перемещение материала, на которое есть РЗ, на штабель, который не является целью	Поднять материал, на которое есть задание Опустить его на штабель, не являющийся штабелем-целью	После опускания материала в гриде РЗ исчезнет активное РЗ на данный материал и появится новое, в котором штабель-цель останется таким же, а штабелем-источником станет штабель, на который материал был опущен В сообщениях появится телеграмма "Вниз" с критическим уровнем, в котором указано, куда перемещён материал, куда должен был быть перемещён, а также информация о	11.03.2016	Пройден получилось только на локалхосте	

			создании нового РЗ.			
	Перемещение материалов без задания	Поднять материал без задания Перенести его на другой штабель	Для перемещённых материалов будет создано рабочее задание со статусом "Завершено", в гриде РЗ это задание не будет показано	11.03.2016	Пройден получилось только на локалхосте	Эти задания может увидеть бригадир в справочнике "Материал"
	Изменение масштаба обзора склада	Нажать несколько раз кнопку "Увеличить" на обзоре склада Нажать несколько раз кнопку "Уменьшить"	При нажатии на "Увеличить" масштаб склада увеличивается, при нажатии на "Уменьшить" - уменьшается	11.03.2016	Пройден	
	Перетаскивание обзора склада	Нажать на любом элементе обзора склада и не отпуская курсора переместить его в другое место	Произойдёт перемещение обзора склада	11.03.2016	Пройден	
	Наличие сообщения о незавершённых заданиях при свёрнутом гриде РЗ, и при отсутствии активных	Включить режим работы крана "ОТСЛЕЖИВАНИЕ ВКЛ" Свернуть грид РЗ	На кнопке "Показать РЗ" появится сообщение о незавершённых заданиях	11.03.2016	Пройден	
	Калибровка вагона	Взять в работу задание на калибровку	Центр вагона окажется под центром крана на обзоре склада	11.03.2016	Пройден получилось только на	

		вагона Переместить кран к реальному месту нахождения вагона Нажать кнопку калибровать	Вагон сменит свой цвет с красного на жёлтый Задание на калибровку вагона исчезнет из грида рабочих заданий В сообщениях появится информация о том, что вагон откалиброван		локалхосте	
	Калибровка штабеля	Взять в работу задание на калибровку штабеля Переместить кран к реальному месту нахождения штабеля Нажать кнопку калибровать	Центр штабеля окажется под центром крана на обзоре склада Задание на калибровку штабеля исчезнет из грида рабочих заданий В сообщениях появится информация о том, что штабель откалиброван	11.03.2016	Пройден получилось только на локалхосте	
	Калибровка штабеля при нахождении крана не над текущим сектором штабеля	Выполнить операции необходимы для калибровки штабеля, находясь на другом секторе	Штабель подсветится красным В сообщениях появится критическая информация о том, что позиция крана не соответствует исходному сектору штабеля	11.03.2016	Пройден получилось только на локалхосте	
	Калибровка вагона при нахождении крана не над рельсом	Выполнить операции необходимы для калибровки штабеля,	В сообщениях появится критическая информация о том, что позиция крана не соответствует	11.03.2016	Пройден получилось только на локалхосте	

		находясь не на рельсе	рельсе			
3	Выход из приложения	Нажать на кнопку "Закрыть"	Программа закрывается	11.03.2016	Пройден	

ПРИЛОЖЕНИЕ 2
Текст программы

ChangeUnitStateView.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Drawing;
using System.Linq;
using System.Reflection;
using System.Windows.Forms;
using System.Windows.Forms.VisualStyles;
using DevExpress.ExpressApp.Win.SystemModule;
using DevExpress.ExpressApp.Win.Templates.ActionContainers;
using DevExpress.XtraLayout;
using DevExpress.XtraLayout.Utils;
using Malahit.MES.BOL.Interface.Enums.References;
using Malahit.UI.Forms.Controls;
using Microsoft.Practices.CompositeUI.SmartParts;
using Microsoft.Practices.ObjectBuilder;

namespace Malahit.MES.UI.Forms.CraneWorkplace.Views.ChangeUnitStateView
{
  public partial class ChangeUnitStateView : CardBase, IChangeUnitStateView,
IComponent
  {
    public ChangeUnitStateView()
    {
      InitializeComponent();
      Init();
    }

    [CreateNew]
    public ChangeUnitStateViewPresenter Presenter
    {
      get { return _presenter; }
      set
```

```
            {
                _presenter = value;
                _presenter.View = this;
            }
        }


        protected override void OnLoad(EventArgs e)
        {
            _presenter.OnViewReady();
            base.OnLoad(e);
        }

        public void ResetBinding()
        {
            _bindingSource.ResetBindings(false);
        }

      public void Init()
      {
          var xOffset = 0;
          //Лукманова А.Р. Используем только два режима!
          foreach (var state in new List<UnitState>(){UnitState.Offline,
UnitState.SemiAuto})
          {
              var newButton = new System.Windows.Forms.Button();
              newButton.Name = state.ToString();
              newButton.Size = new System.Drawing.Size(150, 70);
              newButton.Text = GetEnumDescription(state);
              newButton.Tag = state;
              newButton.Font = new Font("Arial Narrow", 12, FontStyle.Regular);
              newButton.Location = new Point(1 + xOffset, 1);
              newButton.Click += UnitStateClick;

              xOffset += 150;
              panel.Controls.Add(newButton);
          }
      }

      private string GetEnumDescription(UnitState value)
      {
          FieldInfo fieldInfo = value.GetType().GetField(value.ToString());
```

```
            var attributes =
(DescriptionAttribute[])fieldInfo.GetCustomAttributes(typeof(DescriptionAttribute),
false);
            return attributes.Length > 0 ? attributes[0].Description : value.ToString();
        }

        public void UnitStateClick(object sender, EventArgs eventArgs)
        {
          _presenter.SetUnitState((UnitState)(sender as Button).Tag);
          _btOK.Visible = true;
          this._btOK.PerformClick();
          _btOK.Visible = false;
        }


    }
}
```

CraneWorkplaceViewPresenter.cs

```
using System;
using System.Collections.Generic;
using System.Drawing;
using System.Linq;
using System.Windows.Forms;
using System.Windows.Threading;
using Malahit.MES.BOL.Interface;
using Malahit.MES.BOL.Interface.Enums.Manufacture;
using Malahit.MES.Services.Interface.DTO;
using Malahit.MES.Services.Interface.DTO.Common;
using Malahit.MES.Services.Interface.DTO.Manufacture;
using Malahit.MES.Services.Interface.DTO.References;
using Malahit.MES.Services.Interface.Manufacture;
using Malahit.MES.UI.Forms.Common;
using Malahit.MES.UI.Forms.Common.Interface.Services;
using Malahit.MES.UI.Forms.Common.Services;
using Malahit.MES.UI.Forms.Crane;
using Malahit.MES.UI.Forms.Crane.Interface;
using Malahit.MES.UI.Forms.Crane.Interface.Views;
using Malahit.MES.UI.Forms.CraneWorkplace.Services;
using Malahit.MES.UI.Forms.CraneWorkplace.Views.ChangeUnitStateView;
using Malahit.MES.UI.Forms.Material.Interface.PresenterSettings;
using Malahit.MES.UI.Forms.Material.Views.MaterialVisualView;
using Malahit.MES.UI.Forms.TransportJob;
```

```
using Malahit.MES.UI.Forms.Unit.Views.ChangeUnitAvailabilityView;
using Malahit.UI.Forms.Infrastructure.Interface;
using Malahit.UI.Forms.Infrastructure.Interface.Constants;
using Malahit.UI.Forms.Infrastructure.Interface.Services;
using Malahit.UI.Forms.Infrastructure.Library.Services;
using Malahit.UI.Forms.Interfaces;
using Microsoft.Practices.CompositeUI;
using Microsoft.Practices.ObjectBuilder;
using Malahit.MES.BOL.Interface.Enums.References;
using ICraneStatusService =
Malahit.MES.Services.Interface.Manufacture.ICraneStatusService;
using ITransportJobService =
Malahit.MES.Services.Interface.Manufacture.ITransportJobService;
using ITransportService =
Malahit.MES.Services.Interface.References.ITransportService;
using IUnitService = Malahit.MES.Services.Interface.References.IUnitService;
using Malahit.MES.UI.Forms.CraneWorkplace.Interface.Services;

namespace Malahit.MES.UI.Forms.CraneWorkplace
{



  namespace Malahit.MES.UI.Forms.CraneWorkplace
  {
    public class CraneWorkplaceViewPresenter : Presenter<CraneWorkplaceView>,
ICraneAndJobHolder
    {


      [ServiceDependency]
      public Common.Interface.Services.IUnitService UnitService { get; set; }

      [ServiceDependency]
      public Common.Interface.Services.ICraneStatusService CraneStatusService {
get; set; }

      [CreateNew]
      public UserInformer UserInformer { get; set; }

      [ServiceDependency]
      public Common.Interface.Services.ITransportJobService TransportJobService {
get; set; }
```

```
        [ServiceDependency]
        public Common.Interface.Services.IDynamicService DynamicService { get; set;
}

        [ServiceDependency]
        public IMovementService MovementService { get; set; }

        [ServiceDependency]
        public Common.Interface.Services.IMaterialService MaterialService { get; set;
}

        private void OnMessageSended(object sender, EventArgs<DTOBase> e)
        {
          if (e.Data != null)
          {
            MessageListDTO _messages = (e.Data as MessageListDTO);
            View.CraneListMessage(_messages);
          }
        }

        private void OnTransportJobStatusChanged(object sender, EventArgs
eventArgs)
        {
          EnableDisableCalibrateButton();

          ShowCraneViewAndHighlightPiles();
        }

        private void OnEnabledTransportGridButtons()
        {
          _transportJobView.Presenter.GridPresenter.GridEditOptions =
EditOptions.None;
          _transportJobView.Presenter.IsAddButtons = false;
        }

        private const TransportJobStatus ActiveTransportJobStatus =
TransportJobStatus.Active;
        private const TransportJobStatus ShowedTransportJobStatus =
TransportJobStatus.Released;
        private TransportDTO _crane;
        private readonly DispatcherTimer _dispatcherTimer = new DispatcherTimer();
```

```
private readonly DispatcherTimer _stayCraneTimer = new DispatcherTimer();
private int _stayCraneSeconds = 0;

private WarehouseDTO _lastWarehouseUnderTheCrane;
private TransportJobView _transportJobView;
private RouteStepFilter _routeStepFilter;

private bool _autoViewMode = true;

private CraneView _craneView;
private long _oldHallId;



private bool _testIsStarting = false;

public void MoveTest()
{

  if (_craneView != null)
  {


    if (!_testIsStarting)
    {
      _craneView.StartMovingTest();
      View.BtnTest.ForeColor = Color.Green;
    }
    else
    {
      _craneView.StopMovingTest();
      View.BtnTest.ForeColor = Color.Black;

    }


   _testIsStarting = !_testIsStarting;



}
```

```
        }

        public void ShowView()
        {
          //          WorkItem.Services.Get<IWaitService>().OpenWaitDialog("Идет
открытие формы...");

            try
            {


              View = WorkItem.SmartParts.AddNew<CraneWorkplaceView>();
              View.CraneWorkplaceViewPresenter = this;

              View.BtnDown.Enabled = false;
              View.BtnLost.Enabled = false;

              CreateMaterialView();
              GetMaterialView().VerticalScroll.Enabled = false;


              //**************

              var tabPage = new TabPage("Обзор склада");
              tabPage.Name = "tabPageWarehouse";
              tabPage.Controls.Add(_craneView);
              GetMaterialView().tabPanel.TabPages.Add(tabPage);
              //**************


              var transports = new List<TransportDTO>();
              var halls = UnitService.GetAllVisibleHalls();

              foreach (var hall in halls)
              {
                var transportOfHall = CABClientProxy<ITransportService>.Invoke(
                  service => service.GetCranesByHallId(hall.ID.Value))
                  .Select(x =>
                  {
                    x.HallId = hall.ID;
                    return x;
```

```
            _routeStepFilter.HallId = _crane.HallId;


            _transportJobView =
childWorkItem.SmartParts.AddNew<TransportJobView>();
            _transportJobView._transportJobGrid.IsCraneWorkplaceMode = true;
            _transportJobView.SetColumnsForWorkplace();
            _transportJobView.ShouldActivateWaitService = false;

_transportJobView.SetRowHeight(CraneWorkplaceConst.TransportJobRowHeight);

_transportJobView.SetFontSize(CraneWorkplaceConst.TransportJobFontSize);

_transportJobView.SetHeaderFontSize(CraneWorkplaceConst.TransportJobHeaderFon
tSize);
            _transportJobView.SetColumnsWidth();
            _transportJobView._transportJobGrid.HideFilterRow();
            _transportJobView.OnPresenterInitialized +=
OnEnabledTransportGridButtons;
            _transportJobView._transportJobGrid.BarManagerRefresh();
            _transportJobView.Presenter.TransportJobStatusChanged +=
OnTransportJobStatusChanged;
            _transportJobView.Presenter.MessageSended += OnMessageSended;

            View.TaskWorkSpace.Show(_transportJobView);

            _transportJobView.OnPresenterInitialized -=
OnEnabledTransportGridButtons;

            if (transports.Any())
            {
               TabChanged(transports.FirstOrDefault());
               EnableDisableCalibrateButton();
            }



            MoveCraneEvent += MoveCraneHandler;
            StopCraneEvent += StopCraneHandler;
            ChangeAutoViewButtonStyle();
```

```
        }
        finally
        {
            WorkItem.Services.Get<IWaitService>().Close();
        }

    }

    private void EnableDisableCalibrateButton()
    {


        var currentTransportTask = GetCurrentTransportTask().FirstOrDefault();


        if (currentTransportTask == null)
        {
            View.BtnCalibrate.Enabled = false;
            return;

        }


        if (currentTransportTask.Type == MaterialRouteType.Calibration)
        {
            View.BtnCalibrate.Enabled = true;


        }
        else
        {
            View.BtnCalibrate.Enabled = false;
        }

    }


    public void EnableDisableDownButtons()
```

```
        {


            var isUnderPile = IsWarehouseUnderTheCrane(_crane,
_lastWarehouseUnderTheCrane);


            if (isUnderPile == true)
            {

                var materials = MaterialService.GetByUnitId((long)_crane.ID);

                if (materials.Any())
                {
                    View.BtnDown.Enabled = true;
                    View.BtnLost.Enabled = true;
                }
                else
                {
                    View.BtnDown.Enabled = false;
                    View.BtnLost.Enabled = false;
                }
            }
            else
            {
                View.BtnDown.Enabled = false;
                View.BtnLost.Enabled = false;

            }



        }



        public void AutoViewButtonClick()
        {
            _autoViewMode = !_autoViewMode;
            ChangeAutoViewButtonStyle();

        }
```

```
        private void ChangeAutoViewButtonStyle()
        {

            if (_autoViewMode)
                View.BtnAutoView.ForeColor = Color.Green;
            else
                View.BtnAutoView.ForeColor = Color.Red;
        }


        private void Refresh(object sender, EventArgs eventArgs)
        {
            Refresh();
        }

        private void Refresh()
        {
            var tabPanel = GetMaterialView().TabPanel;
            if (tabPanel.SelectedTab.Name == "tabPageWarehouse")
                return;

            if (_crane == null)
                return;


            var oldX = _crane.CoorX;
            var oldY = _crane.CoorY;

            RefreshCraneInfo();

            if (_lastWarehouseUnderTheCrane == null ||
                IsWarehouseUnderTheCrane(_crane, _lastWarehouseUnderTheCrane) !=
true)
            {
                // EnableDisableDownButtons(false);
                GetAndPaintWarehousesUnderTheCrane();
            }
            else if (_lastWarehouseUnderTheCrane != null && _crane.CoorX.HasValue
&& _crane.CoorY.HasValue &&
                     _crane.CoorZ.HasValue)
            {
```

```
                GetMaterialView().SetCranePosition(_crane.CoorX.Value,
_crane.CoorY.Value, _crane.CoorZ.Value);
                // EnableDisableDownButtons(true);
            }

            EnableDisableDownButtons();


            if (oldX == _crane.X && oldY == _crane.Y)
            {
                OnStopCrane(new EventArgs());
            }
            else
            {
                OnMoveCrane(new EventArgs());
            }

            // SetCraneMovementArrows(oldX, oldY, _crane.X, _crane.Y);
        }

        private bool? IsWarehouseUnderTheCrane(TransportDTO transport,
WarehouseDTO warehouse)
        {
            if (transport == null || warehouse == null)
                return null;

            //manapov: здесь отслеживалось что середина находится над штабелем, в
рез. штабель исчезал раньше времени
            //return warehouse.FromCoorX <= transport.CoorX && warehouse.ToCoorX
>= transport.CoorX &&
            //      warehouse.FromCoorY <= transport.CoorY && warehouse.ToCoorY
>= transport.CoorY;

            // сделал пока чтоб всегда показывался, с подстветкой материала ршили
пока не подсвчивать
            return true;



        }

        private void RefreshCraneInfo()
        {
```

```
            var craneStatus = CraneStatusService.GetCraneStatuses(new[] {
 _crane.OriginalName }).SingleOrDefault();

            if (craneStatus != null)
            {
               _crane.CoorX = craneStatus.CoorX;
               _crane.CoorY = craneStatus.CoorY;
               _crane.CoorZ = craneStatus.CoorZ;
            }
         }

         //private void AddTransportJobs(List<DTODynamic> dynamicDTOs)
         //{



         //   var baseTransportJobDTO = new TransportJobDTO
         //   {
         //      Type = MaterialRouteType.MovementByCrane,
         //      TransportJobStatus = TransportJobStatus.Confirmed,
         //      IsAutoCreation = true
         //   };

         //   List<ResultInfo> resultInfos;
         //   List<long> srcAndDstUnit;
         //   MaterialService.CreateTransportJobsForMaterials(baseTransportJobDTO,
dynamicDTOs, out resultInfos,
         //      out srcAndDstUnit);

         //}

         public void SetMaterialsToUnit()
         {


            if (_crane == null)
               return;

            if (_crane.ID == null)
               return;

            var ids = MaterialService.GetByUnitId((long)_crane.ID).Select(x =>
(long)x.ID).ToList();
```

```
            if (!ids.Any())
               return;


            var hallId = _crane.HallId;
            var unit = UnitService.FindUnitByCoords((int)_crane.CoorX,
(int)_crane.CoorY, (long)hallId);

            if (unit == null)
               return;

            var materialFilter = new MaterialFilter
            {
               IDs = ids
            };

            var dynamicDTOs =
DynamicService.GetByParameters(typeof(DTODynamic).AssemblyQualifiedName,
               materialFilter, new List<string> { "ID", "UnitID", "UnitDestID",
"LayerTop" });

            foreach (var dto in dynamicDTOs)
            {
               dto["UnitDestID"] = unit.ID;

            }

            //AddTransportJobs(dynamicDTOs);

            //foreach (var dto in dynamicDTOs)
            //{
            //   dto["UnitID"] = (long)unit.ID;
            //   DynamicService.Update("Material", dto);
            //}


            List<long> srcDstUnitIds;
            List<ResultInfo> resultInfoList;
            MovementService.InnerMoveMaterials(dynamicDTOs);

            GetMaterialView().RefreshMaterials();
```

```
        }

        public void SetMaterialsToCrane()
        {


          if (_crane == null)
            return;

          if (_crane.ID == null)
            return;


          //  return;


          var topMaterialId = GetMaterialView().GetSingleMaterial();

          if (topMaterialId <= 0)
            return;

          var ids = new List<long>() { topMaterialId };//
GetMaterialView().Presenter.SelectedMaterials;

          if (!ids.Any())
            return;

          var materialFilter = new MaterialFilter
          {

            IDs = ids

          };

          var dynamicDTOs =
DynamicService.GetByParameters(typeof(DTODynamic).AssemblyQualifiedName,
              materialFilter, new List<string> { "ID", "UnitID", "UnitDestID",
"LayerTop", "Weight", "MaterialTypeCode" });


          var baseDtoList =
DynamicService.ConvertDynamicDTOList(dynamicDTOs.Cast<DTOBase>().ToList()
, typeof(MaterialDTO));
```

```
        var materialDTOs = baseDtoList.Cast<MaterialDTO>().ToList();

        View.CraneListMessage(GetMaterialView().Validation(materialDTOs));

        if (!dynamicDTOs.Any())
            return;


        foreach (var dto in dynamicDTOs)
        {
            dto["UnitDestID"] = (long)_crane.ID;

        }

        List<long> srcDstUnitIds;
        List<ResultInfo> resultInfoList;
        MovementService.InnerMoveMaterials(dynamicDTOs);

        //AddTransportJobs(dynamicDTOs);

        //foreach (var dto in dynamicDTOs)
        //{
        //    dto["UnitID"] = (long)_crane.ID;
        //    DynamicService.Update("Material", dto);
        //}

        GetMaterialView().RefreshMaterials();


    }

    private void GetAndPaintWarehousesUnderTheCrane()
    {
        if (_crane == null)
            return;



        if (_crane.ID != null)
            GetMaterialView().CraneId = (long)_crane.ID;
```

```
            if (_crane.CoorX == null || _crane.CoorY == null || _crane.CoorZ == null)
            {
                UserInformer.Infrom(@"Не удалось получить координаты крана " +
_crane.OriginalName);
                return;
            }

            var warehousesUnderTheCrane =
UnitService.GetWarehousesUnderThePoint(_crane.CoorX.Value,
                _crane.CoorY.Value);

            if (warehousesUnderTheCrane.Count > 1)
            {
                UserInformer.Infrom(@"Под краном находится сразу несколько
штабелей. Взят первый штабель");
                //TODO Может быть список штабелей?
            }

            var warehouseUnderTheCrane =
warehousesUnderTheCrane.FirstOrDefault();

            _lastWarehouseUnderTheCrane = warehouseUnderTheCrane;

            if (warehouseUnderTheCrane != null)
            {
                //  const float scaleX = 0.2f;
                //  const float scaleY = 0.2f;
                //  var showingParams = new ShowingParams { ScaleX = scaleX, ScaleY
= scaleY, UnitWidth = (int)(GetMaterialView().Width * 0.985 / scaleY), View =
ViewType.Left, ParentWidth = View.Width };
                //  WareHouseReviewViewPresenter.HandleSelectUnitEvent(null, new
SelectUnitEventArgs(warehouseUnderTheCrane.ID.Value)  { ShowingParams =
showingParams });

                if (warehouseUnderTheCrane.ID != null)
                    DrowMaterialView(warehouseUnderTheCrane.ID.Value, null);
            }
            else
            {
                GetMaterialView().ClearElementsPanel();
            }
```

```
            GetMaterialView().SetCranePosition(_crane.CoorX.Value,
_crane.CoorY.Value, _crane.CoorZ.Value);
        }

        /// <summary> Изменяет окрас стрелок-подсказок по движению крана
</summary>
        private void SetCraneMovementArrows(long? oldX, long? oldY, long? newX,
long? newY)
        {
            if ((oldX.HasValue && oldY.HasValue && newX.HasValue &&
newY.HasValue) == false)
                return;

            const int maxMovementPerSecond = 1000;
            ViewType xViewType, xViewTypeForClean;

            if (oldX > newX)
            {
                xViewType = ViewType.Left;
                xViewTypeForClean = ViewType.Right;
            }
            else
            {
                xViewType = ViewType.Right;
                xViewTypeForClean = ViewType.Left;
            }

            int percent = (int)Math.Abs(oldX.Value - newX.Value) * 100 /
maxMovementPerSecond;

            if (percent > 100)
                percent = 100;

            View.SetCraneMovement(percent, xViewType);
            View.SetCraneMovement(0, xViewTypeForClean);

            ViewType yViewType, yViewTypeForClean;

            if (oldY > newY)
            {
                yViewType = ViewType.Back;
                yViewTypeForClean = ViewType.InFront;
```

```
            }
            else
            {
                yViewType = ViewType.InFront;
                yViewTypeForClean = ViewType.Back;
            }

            percent = (int)Math.Abs(oldY.Value - newY.Value) * 100 /
maxMovementPerSecond;

            View.SetCraneMovement(percent, yViewType);
            View.SetCraneMovement(0, yViewTypeForClean);
        }

        private void TabChanged(TransportDTO crane)
        {
            try
            {
                _crane = crane;
                RefreshCraneInfo();
                GetAndPaintWarehousesUnderTheCrane();

                _routeStepFilter.HallId = crane.HallId;
                _routeStepFilter.TransportId = crane.ID;

                RefreshTransportTaskList();

                View.SetCraneUnitState(crane.UnitState);

                ShowCraneViewAndHighlightPiles();
            }
            catch (Exception exception)
            {
                MessageBox.Show(@"Ошибка: " + exception);
                throw;
            }
        }



        private void ShowCraneViewAndHighlightPiles()
        {
```

```
            var tabPage =
GetMaterialView().tabPanel.TabPages.Cast<TabPage>().FirstOrDefault(x => x.Text
== "Обзор склада");

            if (tabPage == null)
            {
               tabPage = new TabPage("Обзор склада");
               GetMaterialView().tabPanel.TabPages.Add(tabPage);
            }


            if (_craneView == null)
            {
               _craneView = WorkItem.SmartParts.AddNew<CraneView>();
               _craneView.Init();
            }

            _craneView.DefaultScaleKoeff = 5;


            if (_oldHallId != _crane.HallId.Value)
            {
               var panel = _craneView.CreatePanel(_crane.HallId.Value);
               _craneView.ElementsPanel = panel.Controls[0] as Panel;
               _craneView.GetAndReprintUnitsOfHall(_crane.HallId.Value, panel, new
List<string>(), new List<long>());

            _oldHallId = _crane.HallId.Value;

            }

            _craneView.ActiveCraneName = Crane.OriginalName;

            var currentTransportTask = GetCurrentTransportTask().FirstOrDefault();
            Job = currentTransportTask;

            tabPage.Controls.Clear();
            tabPage.Controls.Add(_craneView.ElementsPanel);
            _craneView.SetCraneToCenter(Crane.OriginalName);
            _craneView.HighlightPilesInTransportTask(currentTransportTask,
_craneView.ElementsPanel, this); //Активными могут быть только задания с
одинаковыми источниками и целями
```

```
        }

        private void RefreshTransportTaskList()
        {
            WorkItem.Services.Get<IWaitService>().OpenWaitDialog("Обновление
транспортных заданий...");
            try
            {
                _transportJobView.Presenter.GridPresenter.Refresh();
            }
            finally
            {
                WorkItem.Services.Get<IWaitService>().Close();
            }
        }

        private void CreateMaterialView()
        {

            var settings = new MaterialVisualViewPresenterSettings
            {
                HiddenTabs = new List<string> { "tabPageGrid", "tabSettings" }

            };
            var materialWorkItem =
WorkItem.WorkItems.AddNew<ControlledWorkItem<Material.ModuleController>>();
            materialWorkItem.Controller.ShowMaterialVisual(View.MaterialWorkspace,
settings);

        }

        private List<TransportJobDTO> GetCurrentTransportTask()
        {
            var routeFilter = new RouteStepFilter { TransportId = _crane.ID };
            routeFilter.TransportJobStatuses.Add(ActiveTransportJobStatus);
            var routeSteps =
TransportJobService.GetByParameters(typeof(TransportJobDTO).AssemblyQualifiedN
ame,
                routeFilter);
```

```
            return routeSteps;
        }

        private MaterialVisualView GetMaterialView()
        {
            return (MaterialVisualView)View.MaterialWorkspace.ActiveSmartPart;
        }


        private void DrowMaterialView(long unitId, ShowingParams showingParams)
        {

            var materialView = GetMaterialView();
            if (materialView == null)
                return;

            materialView.ClearElementsPanel();


            // получаем юнит по переданному unitId
            var unitDto =

(WarehouseDTO)UnitService.GetByIdEx(typeof(WarehouseDTO).AssemblyQualified
Name, unitId);
            materialView.UnitId = unitId;

            // получаем тип хранения
            var pileMaterialTypeCode = (int)unitDto.MaterialTypeCode;

            // выставляем параметры отображения
            materialView.ShowingParams = showingParams ?? new
ShowingParams(pileMaterialTypeCode);

            // выставляем какие колонки грида и значения будут видны
            materialView.Presenter.FillVisibleColumnSettings(pileMaterialTypeCode);

            // переррисовываем материалы
            materialView.ShowMaterials();

            CraneView.SetAvailableSign(materialView.UnitButton,
materialView.UnitDTO);
        }
```

```
        public new void Dispose()
        {
            _dispatcherTimer.Stop();
            _dispatcherTimer.Tick -= Refresh;

            MoveCraneEvent -= MoveCraneHandler;
            StopCraneEvent -= StopCraneEvent;
            View.CraneWorkplaceViewPresenter = null;
        }

        /// <summary>
        /// активация РЗ - 14.09.2015 удалить после тестирования
        /// </summary>
        public void SetTransportTask()
        {
            var selectedItems = _transportJobView.GetSelectedItems();

            if (selectedItems.Count == 0)
                return;

            foreach (
                var selectedTransportJob in
                    selectedItems.Where(x => x.TransportJobStatus !=
TransportJobStatus.Active))
            {
                selectedTransportJob.TransportJobStatus = ActiveTransportJobStatus;
                selectedTransportJob.TransportId = _crane.ID.Value;
                TransportJobService.Update(selectedTransportJob);
            }

            RefreshTransportTaskList();
            EnableDisableCalibrateButton();
        }

        /// <summary>
        /// Деактивация РЗ - 14.09.2015 удалить после тестирования
        /// </summary>
        public void CancelTransportTask()
        {
            var selectedItems = _transportJobView.GetSelectedItems();

            if (selectedItems.Count == 0)
```

```
                return;

            var countNonActiveTasks = selectedItems.Count(x => x.TransportJobStatus
!= TransportJobStatus.Active);

            if (countNonActiveTasks > 0)
            {
                UserInformer.Infrom(string.Format("Было выбрано {0} заданий,
которые не активны",
                    countNonActiveTasks));
                return;
            }

            foreach (var currentTransportTask in selectedItems)
            {
                currentTransportTask.TransportJobStatus = ShowedTransportJobStatus;
                TransportJobService.Update(currentTransportTask);
            }

            RefreshTransportTaskList();
            EnableDisableCalibrateButton();
        }


        public void CalibrateTask()
        {
            var selectedItems = _transportJobView.GetSelectedItems();

            if (selectedItems.Count != 1)
            {
                UserInformer.Infrom("Должно быть выбрано одно рабочее задание");
                return;
            }

            var currentTransportTask = selectedItems.First();

            if (currentTransportTask.TransportJobStatus != ActiveTransportJobStatus)
            {
                UserInformer.Infrom("Должно быть выбрано активное рабочее
задание");
                return;
            }
```

```
            if (currentTransportTask.Type != MaterialRouteType.Calibration)
            {
              UserInformer.Infrom("Должно быть выбрано задание на калибровку");
              return;
            }

            if (currentTransportTask.Type == MaterialRouteType.Calibration)
            {

              var unitId = currentTransportTask.UnitID;

              var craneStatus = CraneStatusService.GetCraneStatuses(new[] {
_crane.OriginalName }).SingleOrDefault();
              if (craneStatus != null && unitId != null)
              {
                MessageListDTO __messages =
UnitService.SetUnitCoords((long)unitId, craneStatus);
                if (!__messages.Messages.Where(x => x.MessageType ==
MessageType.Error).Any())
                {
                  currentTransportTask.TransportId = _crane.ID;
                  currentTransportTask.TransportJobStatus =
TransportJobStatus.Confirmed;
                  TransportJobService.Update(currentTransportTask);
                  RefreshTransportTaskList();

                  EnableDisableCalibrateButton();
                }

              }
            }
          }



        public void LostMaterial()
        {
          var virtualPile =
UnitService.GetSectorOfLostMaterialsByHall(_crane.HallId.Value);

          if (virtualPile == null)
            throw new Exception("Не удалось определить виртуальный сектор
потерь текущего пролёта");
```

```
            var materialFilter = new MaterialFilter
            {
               UnitId = _crane.ID
            };

            var dynamicDTOs =
DynamicService.GetByParameters(typeof(DTODynamic).AssemblyQualifiedName,
               materialFilter, new List<string> { "ID", "UnitID", "UnitDestID",
"LayerTop" }).OrderByDescending(x => x["LayerTop"]).ThenBy(x =>
x["LayerPosition"]).ToList();

            if (!dynamicDTOs.Any())
               return;

            var topDTO = dynamicDTOs[0];


            topDTO["UnitDestID"] = virtualPile.ID.Value;

            var DTOs = new List<DTODynamic>();

            DTOs.Add(topDTO);


            List<long> srcDstUnitIds;
            List<ResultInfo> resultInfoList;
            MovementService.InnerMoveMaterials(DTOs);




         GetMaterialView().ShowMaterials();
      }

      public void ChangeUnitState()
      {
         var presenter =
this.WorkItem.Items.AddNew<ChangeUnitStateViewPresenter>();
         var unitState = presenter.ShowChangeUnitStateCard(_crane.ID.Value);
```

```
            _crane.UnitState = unitState;
            View.SetCraneUnitState(unitState);
        }

        public void ChangeUnitAvailability(UnitDTO unitDTO, ColorizableButton
unitButton)
        {
            if (unitDTO == null)
            {
                return;
            }
            var changeUnitAvailableViewPresenter =
WorkItem.Items.AddNew<ChangeUnitAvailableViewPresenter>();
            unitDTO =
changeUnitAvailableViewPresenter.SwitchAvailability(unitDTO);
            changeUnitAvailableViewPresenter.ChangePileAvailable(unitDTO);
            CraneView.SetAvailableSign(unitButton, unitDTO);
        }

        public TransportDTO Crane
        {
            get { return _crane; }
        }

        public MaterialRouteStepDTO Job { get; set; }

        public void ChangeView()
        {

            var tabPanel = GetMaterialView().TabPanel;
            var selName = tabPanel.SelectedTab.Name;

            if (selName == "tabPageVisual")
                tabPanel.SelectTab("tabPageWarehouse");
            else if (selName == "tabPageWarehouse")
                tabPanel.SelectTab("tabPageVisual");


          Refresh();
        }


        //реализуем эвенты на движение и останов крана
```

```
// событие кран начал двигаться
public event EventHandler<EventArgs> MoveCraneEvent;

// Wrap event invocations inside a protected virtual method
// to allow derived classes to override the event invocation behavior
protected virtual void OnMoveCrane(EventArgs e)
{
   EventHandler<EventArgs> handler = MoveCraneEvent;

   // Event will be null if there are no subscribers
   if (handler != null)
   {

      // Use the () operator to raise the event.
      handler(this, e);
   }
}


//событие кран остановился
public event EventHandler<EventArgs> StopCraneEvent;

// Wrap event invocations inside a protected virtual method
// to allow derived classes to override the event invocation behavior
protected virtual void OnStopCrane(EventArgs e)
{
   EventHandler<EventArgs> handler = StopCraneEvent;

   // Event will be null if there are no subscribers
   if (handler != null)
   {

      // Use the () operator to raise the event.
      handler(this, e);
   }
}


//методы
```

```
        //обработчик события начала движения крана
        //если находимся на вкладке штабеля, то останавливаем таймер, и
переходим на вкладку обзора склада
        private void MoveCraneHandler(object sender, EventArgs e)
        {
          if (_autoViewMode)
          {

            var tabPanel = GetMaterialView().TabPanel;
            if (tabPanel.SelectedTab.Name == "tabPageVisual")
            {
              if (_stayCraneTimer.IsEnabled)
              {
                _stayCraneTimer.Stop();
                _stayCraneSeconds = 0;

              }
              GetMaterialView().TabPanel.SelectTab("tabPageWarehouse");
            }
          }

        }


        //обработчик события останова крана
        // если находимся на вкладке обзора, то запускаем таймер
        private void StopCraneHandler(object sender, EventArgs e)
        {
          if (_autoViewMode)
          {

            if (_stayCraneTimer.IsEnabled == false)
            {
              _stayCraneSeconds = 0;
              _stayCraneTimer.Interval = new TimeSpan(0, 0, 1);
              _stayCraneTimer.Tick += CheckTimeOnCraneStaying;
              _stayCraneTimer.Start();

            }
          }

        }
```

```
        /// <summary>
        /// проверяем что стоим заданное кол-во времени, если превышаем то,
переключаем вкладку
        /// </summary>
        /// <param name="sender"></param>
        /// <param name="e"></param>
        private void CheckTimeOnCraneStaying(object sender, EventArgs e)
        {
          if (_autoViewMode)
          {

            var materialView = GetMaterialView();

            if (materialView==null)
              return;

            var tabPanel = GetMaterialView().TabPanel;

            if (tabPanel==null)
              return;

            if (tabPanel.SelectedTab.Name == "tabPageWarehouse")
            {
              _stayCraneSeconds++;

              if (_stayCraneSeconds >=
CraneWorkplaceConst.CheckCraneStayingInSeconds)
              {
                GetMaterialView().TabPanel.SelectTab("tabPageVisual");
              }

            }
          }
        }
      }
    }
}
```

ChangeUnitStateViewPresenter.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using System.Windows.Forms;
using Malahit.MES.BOL.Interface.Constants.References;
using Malahit.MES.BOL.Interface.Enums.References;
using Malahit.MES.Services.Interface.DTO.References;
using Malahit.MES.Services.Interface.References;
using Malahit.MES.UI.Forms.Common.Interface.Constants;
using Malahit.MES.UI.Forms.Common.Services;
using Malahit.UI.Forms.Infrastructure.Interface;
using Malahit.UI.Forms.Interfaces;
using Microsoft.Practices.CompositeUI;
using Microsoft.Practices.ObjectBuilder;
using ItemNames = Malahit.UI.Forms.Constants.ItemNames;

namespace Malahit.MES.UI.Forms.CraneWorkplace.Views.ChangeUnitStateView
{
  public class ChangeUnitStateViewPresenter : Presenter<IChangeUnitStateView>
  {
    private ChangeUnitStateView ChangeUnitStateCardView { get; set; }


    [ServiceDependency]
    public Common.Interface.Services.IUnitService UnitService { get; set; }

    private UnitDTO Entity
    {
      get
      {
        return (View as IEntityCard).Entity as UnitDTO;
      }
      set { (View as IEntityCard).Entity = value; }
    }

    public void SetUnitState(UnitState unitState)
    {
      Entity.UnitState = unitState;
    }


    public UnitState ShowChangeUnitStateCard(Int64 unitId)
    {
      ChangeUnitStateCardView =
WorkItem.SmartParts.AddNew<ChangeUnitStateView>();
```

```
            ChangeUnitStateCardView.CardMode = CardMode.Edit;

            ChangeUnitStateCardView.Entity = new UnitDTO();
            ChangeUnitStateCardView.Entity = UnitService.GetById(unitId);
            ChangeUnitStateCardView.OnDialogResultOK += () =>
ChangePileAvailableCardViewOnOnDialogResultOk((UnitDTO)ChangeUnitStateCard
View.Entity);

WorkItem.Workspaces[WorkspaceNames.ModalWindows].Show(ChangeUnitStateCar
dView);
            return (ChangeUnitStateCardView.Entity as UnitDTO).UnitState;
        }

        private void ChangePileAvailableCardViewOnOnDialogResultOk(UnitDTO
unitDTO)
        {
            UnitService.UpdateEx(unitDTO);
        }
    }
}
```

Module.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using Malahit.MES.BOL.Interface.Constants.References;
using Malahit.MES.BOL.Interface.Enums.References;
using Malahit.MES.Services.Interface.DTO.References;
using Malahit.MES.Services.Interface.References;
using Malahit.MES.UI.Forms.Common.Interface.Constants;
using Malahit.MES.UI.Forms.Common.Services;
using Malahit.UI.Forms.Infrastructure.Interface;
using Malahit.UI.Forms.Interfaces;
using Microsoft.Practices.CompositeUI;
using Microsoft.Practices.ObjectBuilder;
using ItemNames = Malahit.UI.Forms.Constants.ItemNames;

namespace Malahit.MES.UI.Forms.CraneWorkplace.Views.ChangeUnitStateView
{
    public class ChangeUnitStateViewPresenter : Presenter<IChangeUnitStateView>
    {
```

```
        private ChangeUnitStateView ChangeUnitStateCardView { get; set; }


        [ServiceDependency]
        public Common.Interface.Services.IUnitService UnitService { get; set; }

        private UnitDTO Entity
        {
            get
            {
                return (View as IEntityCard).Entity as UnitDTO;
            }
            set { (View as IEntityCard).Entity = value; }
        }

        public void SetUnitState(UnitState unitState)
        {
            Entity.UnitState = unitState;
        }


       public UnitState ShowChangeUnitStateCard(Int64 unitId)
       {
            ChangeUnitStateCardView =
 WorkItem.SmartParts.AddNew<ChangeUnitStateView>();
            ChangeUnitStateCardView.CardMode = CardMode.Edit;

            ChangeUnitStateCardView.Entity = new UnitDTO();
            ChangeUnitStateCardView.Entity = UnitService.GetById(unitId);
            ChangeUnitStateCardView.OnDialogResultOK += () =>
 ChangePileAvailableCardViewOnOnDialogResultOk((UnitDTO)ChangeUnitStateCard
 View.Entity);

 WorkItem.Workspaces[WorkspaceNames.ModalWindows].Show(ChangeUnitStateCar
 dView);
            return (ChangeUnitStateCardView.Entity as UnitDTO).UnitState;
       }

      private void ChangePileAvailableCardViewOnOnDialogResultOk(UnitDTO
 unitDTO)
       {
            UnitService.UpdateEx(unitDTO);
       }
```

```
    }
}
```

CraneWorkplaceConst.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Malahit.MES.UI.Forms.CraneWorkplace
{
    public static class CraneWorkplaceConst
    {
        #region TransportJob

        public static readonly Int32 TransportJobFontSize = 14;
        public static readonly Int32 TransportJobHeaderFontSize = 12;
        public static readonly Int32 TransportJobRecordCount = 5;
        public static readonly Int32 TransportJobRowHeight = 70;

        #endregion

        #region Basic

        public static readonly Int32 CraneBaseFontSize = 12;
        public static readonly Int32 CraneBaseEditTextHeight = 30;
        public static readonly Int32 CraneBaseButtonHeight = 50;

        #endregion

        // через сколько секунд после останова кран считается остановившимся
        public static readonly Int32 CheckCraneStayingInSeconds = 5;

    }
}
```

ModuleController.cs

```
using System;
using Malahit.MES.UI.Forms.Crane;
using
Malahit.MES.UI.Forms.CraneWorkplace.Malahit.MES.UI.Forms.CraneWorkplace;
using Malahit.MES.UI.Forms.Material.Views.MaterialVisualView;
```

```
using Malahit.UI.Forms.Infrastructure.Interface;
using Malahit.UI.Forms.Infrastructure.Interface.Constants;
using Microsoft.Practices.CompositeUI;
using Microsoft.Practices.CompositeUI.SmartParts;
using Microsoft.Practices.ObjectBuilder;

namespace Malahit.MES.UI.Forms.CraneWorkplace
{
   public class Module : ModuleInit
   {
      private readonly WorkItem _rootWorkItem;

      [InjectionConstructor]
      public Module([ServiceDependency] WorkItem rootWorkItem)
      {
         _rootWorkItem = rootWorkItem;
      }

      public override void Load()
      {
         base.Load();

         var workItem =
_rootWorkItem.WorkItems.AddNew<ControlledWorkItem<ModuleController>>();
         workItem.Controller.Run();
      }
   }

   public class ModuleController : WorkItemController
   {
      public override void Run()
      {
         Init();
         BuildMenuItem = (mi, item) =>
ActionCatalogService.RegisterActionImplementation(item.MenuPath,
OnMenuClickAction);
         BuidModuleMenu(false);

         WorkItem.Workspaces[WorkspaceNames.MainWorkspace].SmartPartClosing
+= DisposeView;
      }

      public void OnMenuClickAction(Object caller, Object target)
```

```
        {
            var craneWorkplaceViewPresenter =
 WorkItem.Items.AddNew<CraneWorkplaceViewPresenter>();
            craneWorkplaceViewPresenter.ShowView();
        }

        private static void DisposeView(object sender, WorkspaceCancelEventArgs args)
        {
            var craneWorkplaceView = args.SmartPart as CraneWorkplaceView;

            if (craneWorkplaceView != null)
            {

                var materialView =
 craneWorkplaceView.MaterialWorkspace.ActiveSmartPart as MaterialVisualView;

                if (materialView == null)
                    return;

                materialView.Dispose();

                craneWorkplaceView.CraneWorkplaceViewPresenter.Dispose();
                craneWorkplaceView.Dispose();
            }



        }
    }
}
```

CraneWorkplaceView.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Drawing;
using System.Drawing.Drawing2D;
using System.Linq;
using System.Reflection;
using System.Windows.Forms;
using Malahit.MES.BOL.Interface.Enums.References;
using Malahit.MES.Services.Interface.DTO.Common;
using Malahit.MES.Services.Interface.DTO.References;
```

```
using Malahit.MES.UI.Forms.Common;
using Malahit.MES.UI.Forms.Crane.Interface;
using
Malahit.MES.UI.Forms.CraneWorkplace.Malahit.MES.UI.Forms.CraneWorkplace;
using Malahit.MES.UI.Forms.Material.Constants;
using Malahit.MES.UI.Forms.Material.Views.MaterialVisualView;
using Microsoft.Practices.CompositeUI.SmartParts;
using Microsoft.Practices.CompositeUI.WinForms;
using ContentAlignment = System.Drawing.ContentAlignment;

namespace Malahit.MES.UI.Forms.CraneWorkplace
{
  public partial class CraneWorkplaceView : UserControl, ISmartPartInfoProvider
//,IWorkSpaceHolder,
  {
    public CraneWorkplaceView()
    {
      Dock = DockStyle.Fill;
      InitializeComponent();
      SetDoubleBuffered(_splitContainerVert);

      // решает проблему выдачи ошибки при повторном открытии формы
      TaskWorkSpace.Name = string.Format("DeckMaterialWorkSpace{0}",
_num++);
    }

    public Button BtnDown
    {
      get { return btn_DownMaterial; }
      set { btn_DownMaterial = value; }
    }


    public Button BtnLost
    {
      get { return _lostMaterialButton; }
      set { _lostMaterialButton = value; }
    }

    public Button BtnAutoView
    {
      get { return btn_AutoView; }
      set { btn_AutoView = value; }
```

```
        }

        public Button BtnTest
        {
            get { return btn_test; }
            set { btn_test = value; }
        }

        public Button BtnCalibrate
        {
            get { return btn_Calibrate; }
            set { btn_Calibrate = value; }
        }

        private static int _num;
        public Button selectedCrane;

        public void Init(List<TransportDTO> transports)
        {

            var panel = _splitContainer5.Panel1;

            int xOffset = 3;
            int yOffset = 3;
            int buttonWidth = 80;
            int buttonHeight = 50;
            var hallIds = transports.Select(x => x.HallId.Value).Distinct().ToList();
            var halls = CraneWorkplaceViewPresenter.UnitService.GetByIdList(hallIds);
            foreach (var hall in halls)
            {
                panel.Controls.Add(new Label { Text = hall.Name,  TextAlign =
ContentAlignment.MiddleCenter,  Location = new Point(xOffset, buttonHeight + 3),
Size = new Size(buttonWidth * 3, buttonHeight/2) });

                foreach (var transport in transports.Where(x => x.HallId ==
hall.ID).ToList())
                {
                    var button = new Button { Text = transport.ShortName, Tag = transport,
Location = new Point(xOffset, yOffset), Size = new Size(buttonWidth, buttonHeight),
Font = new Font("Arial Narrow", 16, FontStyle.Regular) };
                    button.Click += TabControlOnSelectedIndexChanged;

                    panel.Controls.Add(button);
```

```
                xOffset += buttonWidth + 4;
                selectedCrane = selectedCrane ?? button;
            }
            xOffset += buttonWidth / 2;
        }

        _splitContainerWarehouse.Panel2.Controls.Add(_materialWorkspace);
        selectedCrane.PerformClick();
    }

    public event Action<TransportDTO> OnTabChanged;

    private void TabControlOnSelectedIndexChanged(object sender, EventArgs
eventArgs)
    {
        RedrawSelectedButton(sender as Button);
        var tabPage = (Control)sender;

        var onTabChanged = OnTabChanged;
        if (onTabChanged != null)
            onTabChanged((TransportDTO)tabPage.Tag);
    }

    private void SetDoubleBuffered(object control)
    {
        control.GetType().GetProperty("DoubleBuffered", BindingFlags.Instance |
BindingFlags.NonPublic);
    }

    public new void Dispose()
    {
        OnTabChanged = null;
        base.Dispose();
    }

    public ISmartPartInfo GetSmartPartInfo(Type smartPartInfoType)
    {
        var result = (ISmartPartInfo)Activator.CreateInstance(smartPartInfoType);
        result.Title = "Рабочее место крановщика";
        result.Description = result.Title;
        var info = result as WindowSmartPartInfo;
        if (info != null)
            info.WindowState = FormWindowState.Maximized;
```

```
            return result;
        }

        private readonly DeckWorkspace _materialWorkspace = new DeckWorkspace {
Dock = DockStyle.Fill };
        public CraneWorkplaceViewPresenter CraneWorkplaceViewPresenter { get; set; }
        public IWorkspace MaterialWorkspace { get { return _materialWorkspace; } }
        public DeckWorkspace TaskWorkSpace{ get { return _taskWorkSpace; }}

        private int _width, _height;
        private int _offsetY;
        private int _leftX, _rightX, _downY, _upY;
        private Control _leftIndicatorControl, _rightIndicatorControl,
_upIndicatorControl, _downIndicatorControl;
        private const int LineWidth = 3;

        private void SetArrowDirection(ViewType viewType, GraphicsPath graphicsPath)
        {
            switch (viewType)
            {
                case ViewType.Left:
                    {
                        var rotation = 90;
                        var rotationPoint = new PointF((int)(_width / 2), (int)(_width / 2));
                        var rotateMatrix = new Matrix();
                        rotateMatrix.RotateAt(rotation, rotationPoint);
                        graphicsPath.Transform(rotateMatrix);

                        var translateMatrix = new Matrix();
                        translateMatrix.Translate(_offsetY, 0);
                        graphicsPath.Transform(translateMatrix);
                        break;
                    }
                case ViewType.InFront:
                    {
                        var rotation = 180;
                        var rotationPoint = new PointF((int)(_width / 2), (int)(_height / 2));
                        var rotateMatrix = new Matrix();
                        rotateMatrix.RotateAt(rotation, rotationPoint);
                        graphicsPath.Transform(rotateMatrix);

                        break;
                    }
```

```
            case ViewType.Right:
                {
                    var rotation = 270;
                    var rotationPoint = new PointF((int)(_width / 2), (int)(_width / 2));
                    var rotateMatrix = new Matrix();
                    rotateMatrix.RotateAt(rotation, rotationPoint);
                    graphicsPath.Transform(rotateMatrix);

                    break;
                }
        }
    }

    public void SetCraneMovement(int percent, ViewType viewType)
    {
        var indicatorGraphicsPath = new GraphicsPath();

        Control indicatorControl = null;

        switch (viewType)
        {
            case ViewType.Left:
            {
                indicatorControl = _leftIndicatorControl;
                break;
            }
            case ViewType.Right:
            {
                indicatorControl = _rightIndicatorControl;
                break;
            }
            case ViewType.InFront:
            {
                indicatorControl = _upIndicatorControl;
                break;
            }
            case ViewType.Back:
            {
                indicatorControl = _downIndicatorControl;
                break;
            }
        }
```

```
            indicatorGraphicsPath.AddPolygon(new[] { new Point(_leftX, _downY +
LineWidth), new Point(_rightX, _downY + LineWidth), new Point(_rightX, _upY *
percent / 100), new Point(_leftX, _upY * percent / 100) });

            SetArrowDirection(viewType, indicatorGraphicsPath);

            indicatorControl.Region = new Region(indicatorGraphicsPath);
            indicatorGraphicsPath.Dispose();
        }



        // Калибровка
        private void btn_Cakibrate_Click(object sender, EventArgs e)
        {
            CraneWorkplaceViewPresenter.CalibrateTask();
        }

        private void LostMaterialButtonClick(object sender, EventArgs e)
        {
            CraneWorkplaceViewPresenter.LostMaterial();
        }

        public void SetCraneUnitState(UnitState unitState)
        {
            _craneUnitStateButton.Text = GetEnumDescription(unitState);
        }

        private string GetEnumDescription(UnitState value)
        {
            FieldInfo fieldInfo = value.GetType().GetField(value.ToString());
            var attributes =
(DescriptionAttribute[])fieldInfo.GetCustomAttributes(typeof(DescriptionAttribute),fal
se);
            return attributes.Length > 0 ? attributes[0].Description : value.ToString();
        }


        private void _craneUnitStateButton_Click(object sender, EventArgs e)
        {
            CraneWorkplaceViewPresenter.ChangeUnitState();
        }
```

```
        private void btn_DownMaterial_Click(object sender, EventArgs e)
        {
            CraneWorkplaceViewPresenter.SetMaterialsToUnit();
        }

        private void btn_UpMaterial_Click(object sender, EventArgs e)
        {
            CraneWorkplaceViewPresenter.SetMaterialsToCrane();

        }
        private void changeUnitAvailabilityButton_Click(object sender, EventArgs e)
        {
            var unitDTO = (MaterialWorkspace.ActiveSmartPart as
MaterialVisualView).UnitDTO;
            var unitButton = (MaterialWorkspace.ActiveSmartPart as
MaterialVisualView).ElementUnit as ColorizableButton;
            CraneWorkplaceViewPresenter.ChangeUnitAvailability(unitDTO, unitButton);
        }


    public void CraneListMessage(MessageListDTO messages)
    {

        ilbMessageList.Items.Clear();
        if (messages != null)
        {
            if (messages.Messages != null)
            {
                foreach (var _mes in messages.Messages)
                {
                    CraneMessage(_mes.MessageType, _mes.MessageText);
                }
            }
        }
    }

    public void CraneMessage(MessageType typeMessage, string message)
    {
        ilbMessageList.Items.Add(message, (int) typeMessage);
    }
```

```
        public void RedrawSelectedButton(Button selectedButton)
        {
            selectedButton.FlatStyle = FlatStyle.Flat;
            selectedButton.FlatAppearance.BorderColor = MaterialColors.Selected;
            selectedButton.FlatAppearance.BorderSize = 3;
            if (selectedCrane != null && selectedCrane!= selectedButton)
                selectedCrane.FlatStyle = FlatStyle.Standard;
            selectedCrane = selectedButton;
        }

        private void _taskWorkSpace_Click(object sender, EventArgs e)
        {

        }

        private void btn_ChangeView_Click(object sender, EventArgs e)
        {
            CraneWorkplaceViewPresenter.ChangeView();
        }

        private void btn_AutoView_Click(object sender, EventArgs e)
        {
            CraneWorkplaceViewPresenter.AutoViewButtonClick();
        }

        private void btn_test_Click(object sender, EventArgs e)
        {
         CraneWorkplaceViewPresenter.MoveTest();
        }


    }
}
```

UserInformer.cs

```
using System.Windows.Forms;

namespace Malahit.MES.UI.Forms.CraneWorkplace
{
    public class UserInformer
    {
        public void Infrom(string text)
        {
```

```
            MessageBox.Show(text);
        }
    }
}
```

IChangeUnitStateView.cs

```
using Microsoft.Practices.CompositeUI.SmartParts;

namespace Malahit.MES.UI.Forms.CraneWorkplace.Views.ChangeUnitStateView
{
    public interface IChangeUnitStateView
    {
        void ResetBinding();
    }
}
```

MovementService.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Malahit.MES.BOL.Interface;
using Malahit.MES.BOL.Interface.Enums.Manufacture;
using Malahit.MES.Services.Interface.DTO;
using Malahit.MES.Services.Interface.DTO.Manufacture;
using Malahit.MES.UI.Forms.CraneWorkplace.Interface.Services;
using Microsoft.Practices.CompositeUI;
using Malahit.MES.UI.Forms.Common.Interface.Services;

namespace Malahit.MES.UI.Forms.CraneWorkplace.Services
{
    [Service(typeof(IMovementService), AddOnDemand = true)]
    public class MovementService:IMovementService
    {
        [ServiceDependency]
        public IMaterialUnloadingService MaterialUnloadingService { get; set; }

        public void InnerMoveMaterials(IList<DTODynamic> dynamicData)
        {
            var unitGroupedData = dynamicData.GroupBy(x =>
x.Properties["UnitDestID"].Value);
```

```
            // иточник у всех материалов один
            long srcUnitId = dynamicData[0].Properties["UnitID"].Value;

            foreach (var group in unitGroupedData)
            {

                var materialIds = group.OrderByDescending(x =>
x.Properties["LayerTop"].Value).Select(x => (long)x.ID).ToList();
                var destUnitId = group.Key;

                List<long> srcAndDstUnit;
                MaterialBulkOperationInfoDTO operationInfo;

                MaterialUnloadingService.MoveMaterials(materialIds, (Int64)destUnitId, out
operationInfo, out srcAndDstUnit);
            }
        }
    }
}
```

Resources.Designer.cs

```
//------------------------------------------------------------------------------
// <auto-generated>
//     This code was generated by a tool.
//     Runtime Version:4.0.30319.34014
//
//     Changes to this file may cause incorrect behavior and will be lost if
//     the code is regenerated.
// </auto-generated>
//------------------------------------------------------------------------------

namespace Malahit.MES.UI.Forms.CraneWorkplace.Properties {
    using System;


    /// <summary>
    ///   A strongly-typed resource class, for looking up localized strings, etc.
    /// </summary>
    // This class was auto-generated by the StronglyTypedResourceBuilder
    // class via a tool like ResGen or Visual Studio.
    // To add or remove a member, edit your .ResX file then rerun ResGen
    // with the /str option, or rebuild your VS project.
```

```
[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Resources.Tool
s.StronglyTypedResourceBuilder", "4.0.0.0")]
    [global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
    [global::System.Runtime.CompilerServices.CompilerGeneratedAttribute()]
    internal class Resources {

        private static global::System.Resources.ResourceManager resourceMan;

        private static global::System.Globalization.CultureInfo resourceCulture;


[global::System.Diagnostics.CodeAnalysis.SuppressMessageAttribute("Microsoft.Perf
ormance", "CA1811:AvoidUncalledPrivateCode")]
        internal Resources() {
        }

        /// <summary>
        ///   Returns the cached ResourceManager instance used by this class.
        /// </summary>

[global::System.ComponentModel.EditorBrowsableAttribute(global::System.Compone
ntModel.EditorBrowsableState.Advanced)]
        internal static global::System.Resources.ResourceManager ResourceManager {
            get {
                if (object.ReferenceEquals(resourceMan, null)) {
                    global::System.Resources.ResourceManager temp = new
global::System.Resources.ResourceManager("Malahit.MES.UI.Forms.CraneWorkplace
.Properties.Resources", typeof(Resources).Assembly);
                    resourceMan = temp;
                }
                return resourceMan;
            }
        }

        /// <summary>
        ///   Overrides the current thread's CurrentUICulture property for all
        ///   resource lookups using this strongly typed resource class.
        /// </summary>

[global::System.ComponentModel.EditorBrowsableAttribute(global::System.Compone
ntModel.EditorBrowsableState.Advanced)]
        internal static global::System.Globalization.CultureInfo Culture {
```

```
            get {
                return resourceCulture;
            }
            set {
                resourceCulture = value;
            }
        }
    }
}
```

ПРИЛОЖЕНИЕ 3
Инструкция пользователю

1.Общие сведения

Данное руководство пользователя описывает интерфейс работы Рабочего Места Крановщика, без описания реализации функций.

2.Назначение системы

Данная система разработана для отслеживания перемещения материалов по цеху.

3.Основные требования при работе с системой

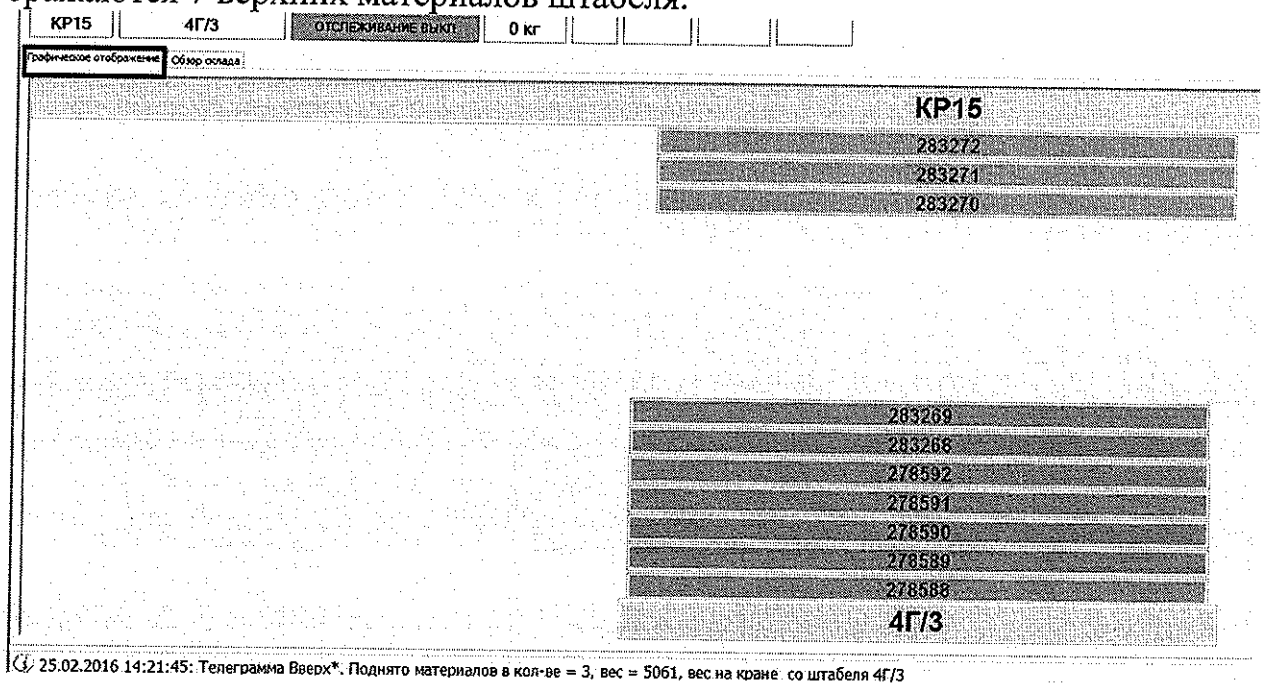
При работе с системой WTMS важно понимать следующее:

- Система работает с материалами, которые были закачены из КИС или созданными бригадиром (они называются виртуальными и подкрашиваются розовым цветом).
- Все перемещения материалов известны, так как происходят по рабочим заданиям, которые либо берутся в работу и выполняются, либо создаются за кадром при перемещении материалов в режиме отслеживания крана.
- Для работы системы необходимо, чтобы кран был откалиброван и места хранения – штабеля, были размечены и откалиброваны (каждому штабелю в системе должны соответствовать его координаты в складе).

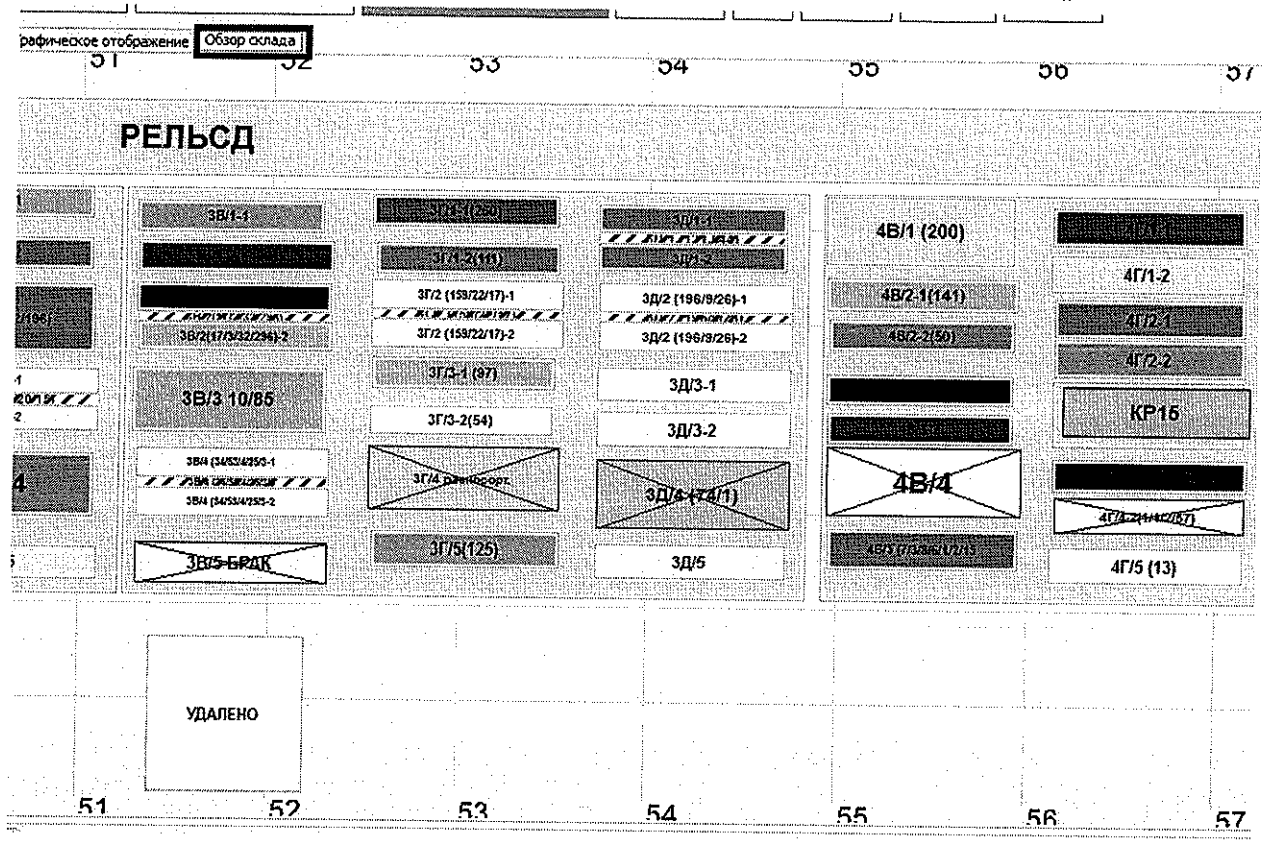
4.Описание рабочего места крановщика

Рабочее Место Крановщика (РМК) состоит из двух вкладок: «Графическое отображение» и «Обзор склада».

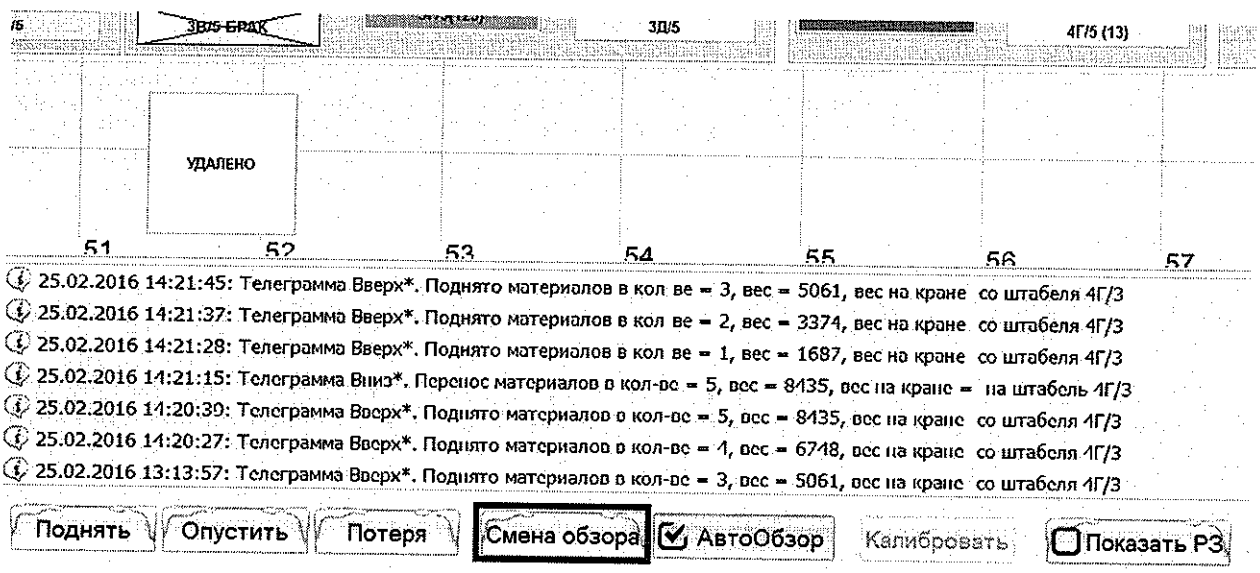
На вкладке «Графическое отображение» отображается штабель, над которым сейчас находится кран. Причём, в «Графическом отображении» отображаются 7 верхних материалов штабеля.



На вкладе «Обзор склада» отображается схема пролёта, в котором находится кран. Цвета штабелей указывают на степень заполненности штабеля материалами.

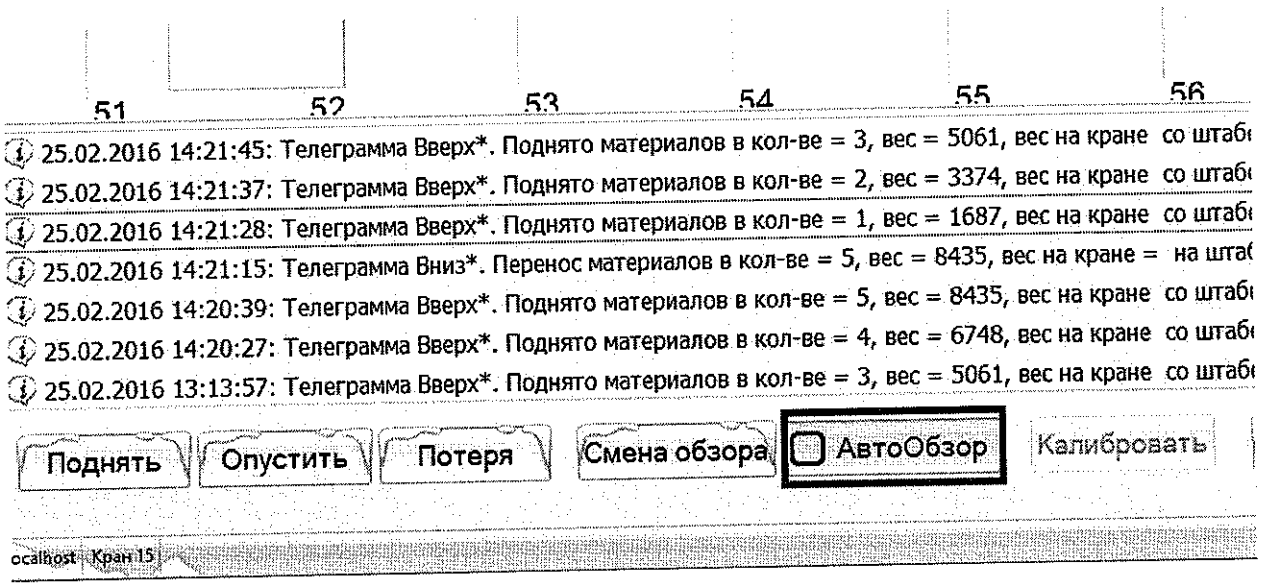


Переход между вкладками можно выполнить с помощью нажатия на кнопку «Смена обзора», которая находится в нижней части экрана.



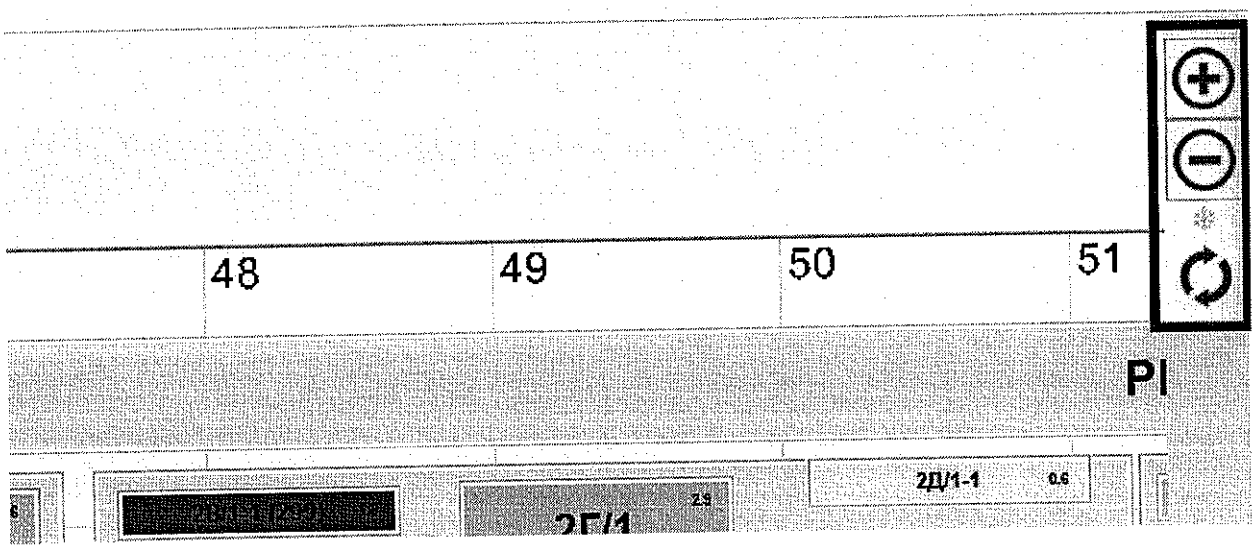
Если на кнопке «АвтоОбзор» выставлена галочка, то при остановке крана над штабелем система автоматически будет переключать вкладку с «Обзор склада» на вкладку «Графическое отображение», где будет отображаться штабель, над которым остановился кран.

Если на кнопке «АвтоОбзор» нет галочки, автоматического переключения между вкладками происходить не будет. Галочка выставляется и убирается путём нажатия на эту кнопку.



На вкладке «Обзор склада» есть возможность увеличения и уменьшения масштаба отображения схемы склада. Для этого в правой стороне обзора склада есть кнопки увеличить  и уменьшить  .

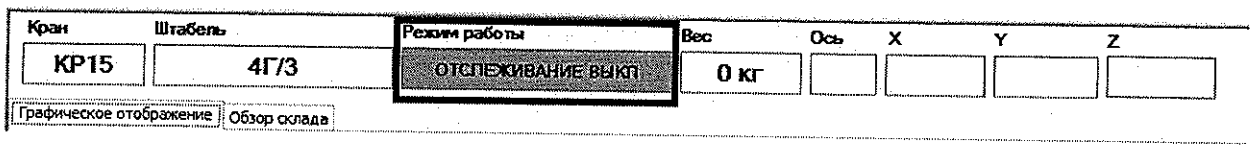
Также, если возникнет необходимость обновить отображение схемы склада, необходимо нажать кнопку обновления  .



5. Информационная панель с параметрами крана

В верхней части экрана РМК расположена информационная панель, на которой отображается имя крана, наименование штабеля, над которым он сейчас

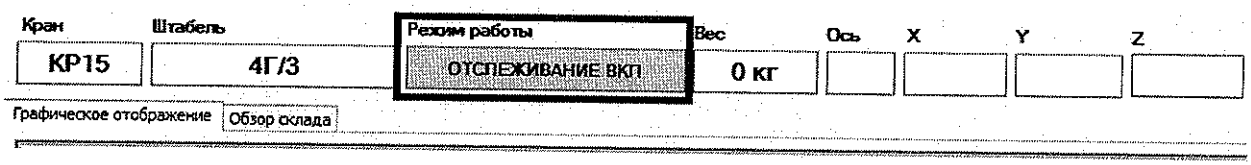
находится, режим работы крана, вес, поднятый краном, ось?, а также координаты крана в пролёте.



Режим работы крана бывают двух видов: «ОТСЛЕЖИВАНИЕ ВЫКЛ» и «ОТСЛЕЖИВАНИЕ ВКЛ».

Если режим работы «ОТСЛЕЖИВАНИЕ ВЫКЛ», то перемещения материалов не фиксируются системой, работа идёт вчёрную.

Если режим крана «ОТСЛЕЖИВАНИЕ ВКЛ», то перемещение материалов отслеживается.



При работе крана в этом режиме перемещения материалов могут осуществляться двумя способами:

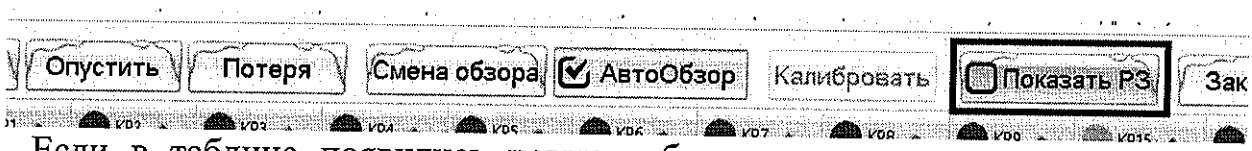
1. По рабочим заданиям, созданным бригадиром пролёта, которые берутся в работу.

2. Перемещения без взятия в работу рабочих заданий. В этом случае перемещения материалов фиксируются системой «за кадром», по телеграммам, поступающим от крана.

6. Работа в режиме крана «ОТСЛЕЖИВАНИЕ ВКЛ»

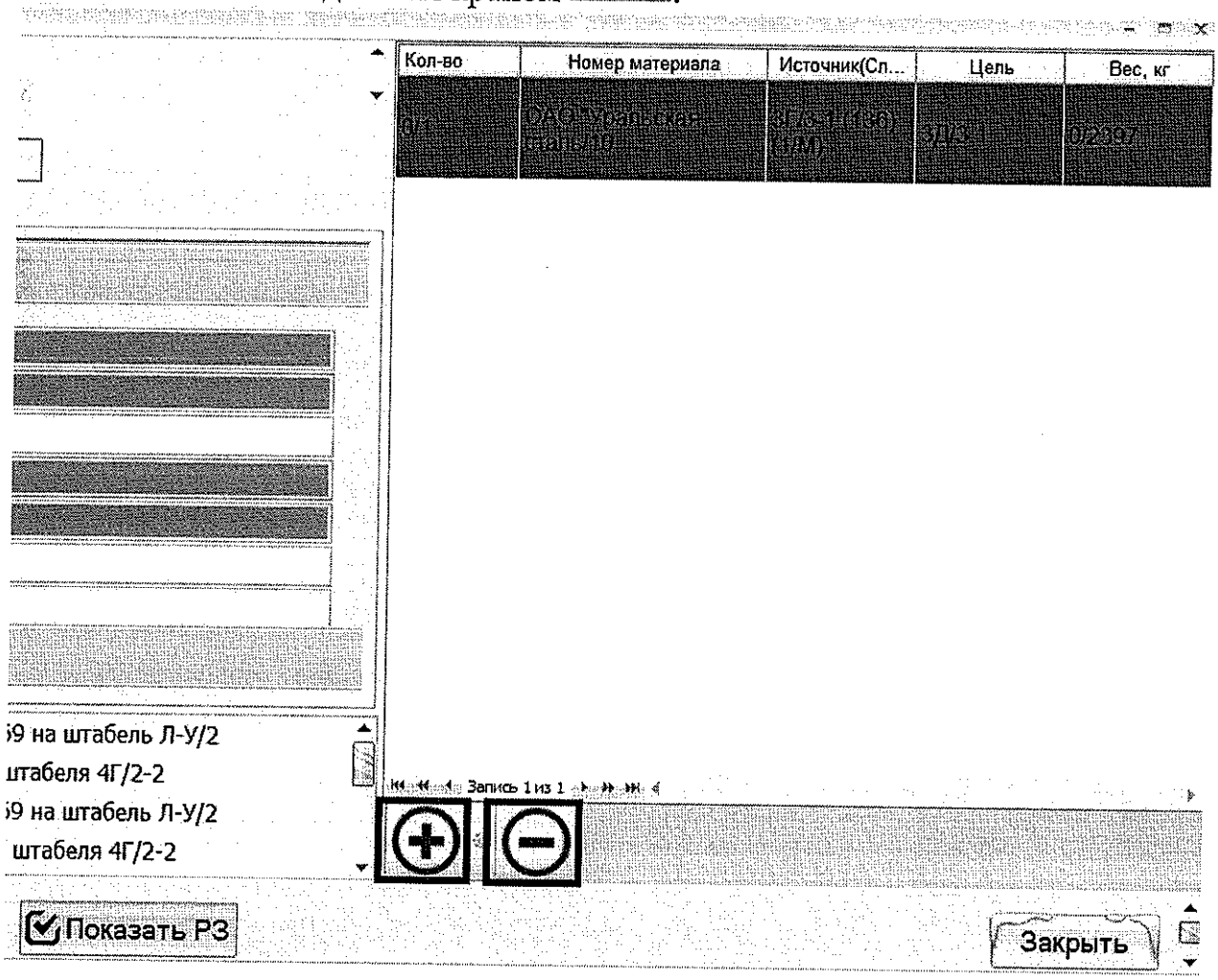
6.1.Работа по рабочим заданиям, создаваемым бригадиром

Если перемещение материалов выполняется по рабочим заданиям, созданным бригадиром, то для этого необходимо взять задание в работу. Для этого в РМК есть таблица с рабочими заданиями (Таблица РЗ). По умолчанию, эта таблица скрыта, чтобы увидеть её или скрыть (если панель с рабочими заданиями открыта), необходимо нажать на кнопку «Показать РЗ».

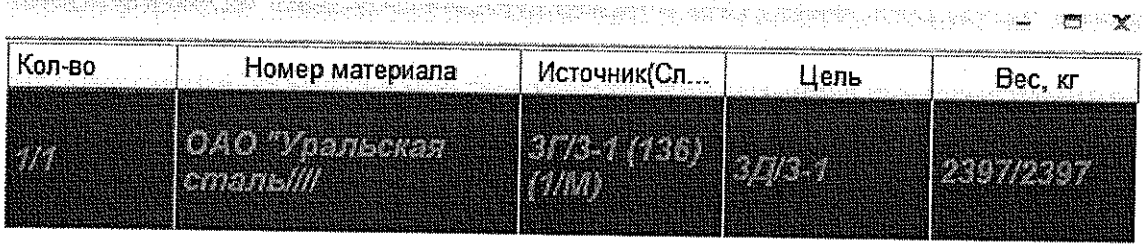


Если в таблице появились новые рабочие задания, а она находится в свёрнутом виде, кнопка изменяет свой цвет.

В таблице рабочих заданий есть кнопки для взятия РЗ в работу  и отмены выполнения РЗ данным краном  .

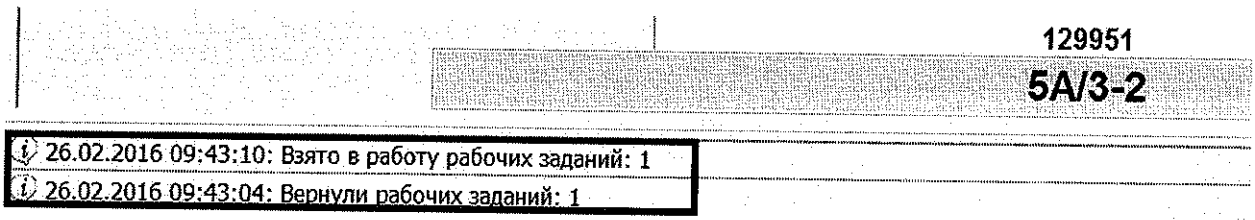


При взятии задания в работу оно меняет свой цвет на оранжевый.

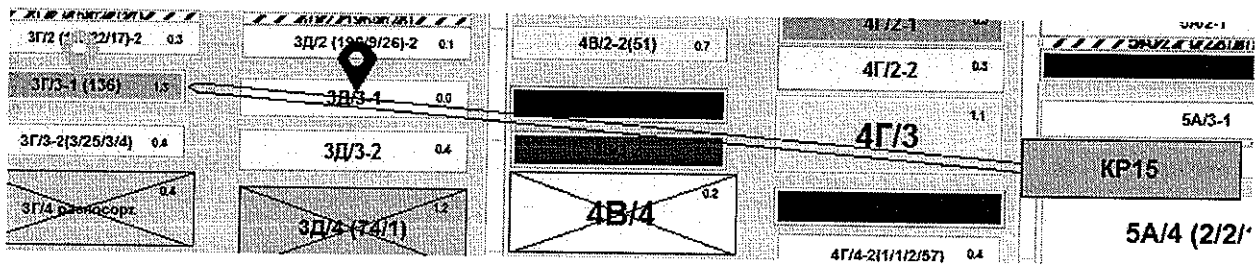


При возвращении задания меняет цвет на чёрный.

Взятие задания в работу и его возвращение сопровождается появлением сообщений в списке сообщений на графическом отображении штабеля.



При перемещении материала по РЗ на вкладке «Обзор склада» появляются подсказки для движения крана: бирюзовый маркер обозначает место, где лежит материал, который нужно переместить, красный маркер-место куда необходимо перенести материал. Также от крана, взявшего задание в работу, до текущего места назначения проводится стрелка для указания направления движения.

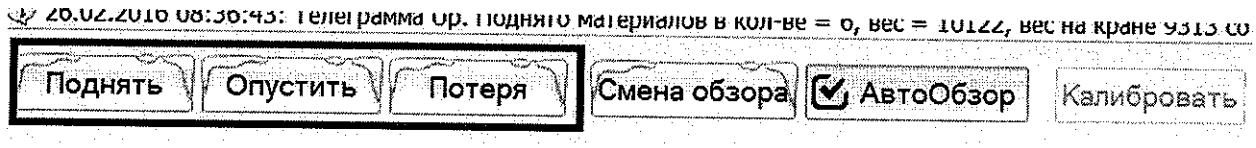


6.2. Работа без рабочих заданий

Если в режиме работы крана с отслеживанием выполняются перемещения материалов без РЗ, то эти перемещения всё равно отслеживаются, так как рабочие задания создаются в системе за кадром по данным, которые отправляются краном при поднятии и опускании материала, эти задания не появляются в таблице РЗ, но фиксируются в системе, по ним также выводятся сообщения.

7. Поднятие, опускание и потеря материала

Находясь на вкладке «Графическое отображение» можно видеть сколько материалов сейчас находится на кране. Эта информация приходит в систему WTMS по телеграмме, отправляемой краном, когда кран выполняет поднятие или опускание материала. В случае, если отображение материалов на кране не соответствует реальности, необходимо откорректировать количество материалов на кране с помощью кнопок «Поднять», «Опустить» и «Потеря», находящихся в нижней части экрана.

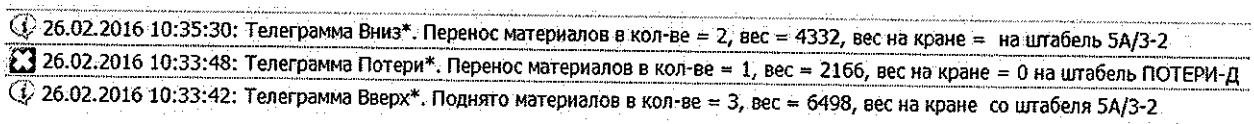


Нажатие кнопки «Поднять» перемещает в системе один материал со штабеля на нижний слой крана. То есть, если необходимо поднять 3 материала на кран, нужно три раза нажать кнопку «Поднять».

Нажатие кнопки «Опустить» перемещает в системе все листы с крана на штабель.

В случае, если во время перемещения произошла потеря материала, а система продолжает отображать потерянный материал на кране, необходимо нажать кнопку «Потеря», которая в системе перемещает один последний материал с крана на штабель «Потери».

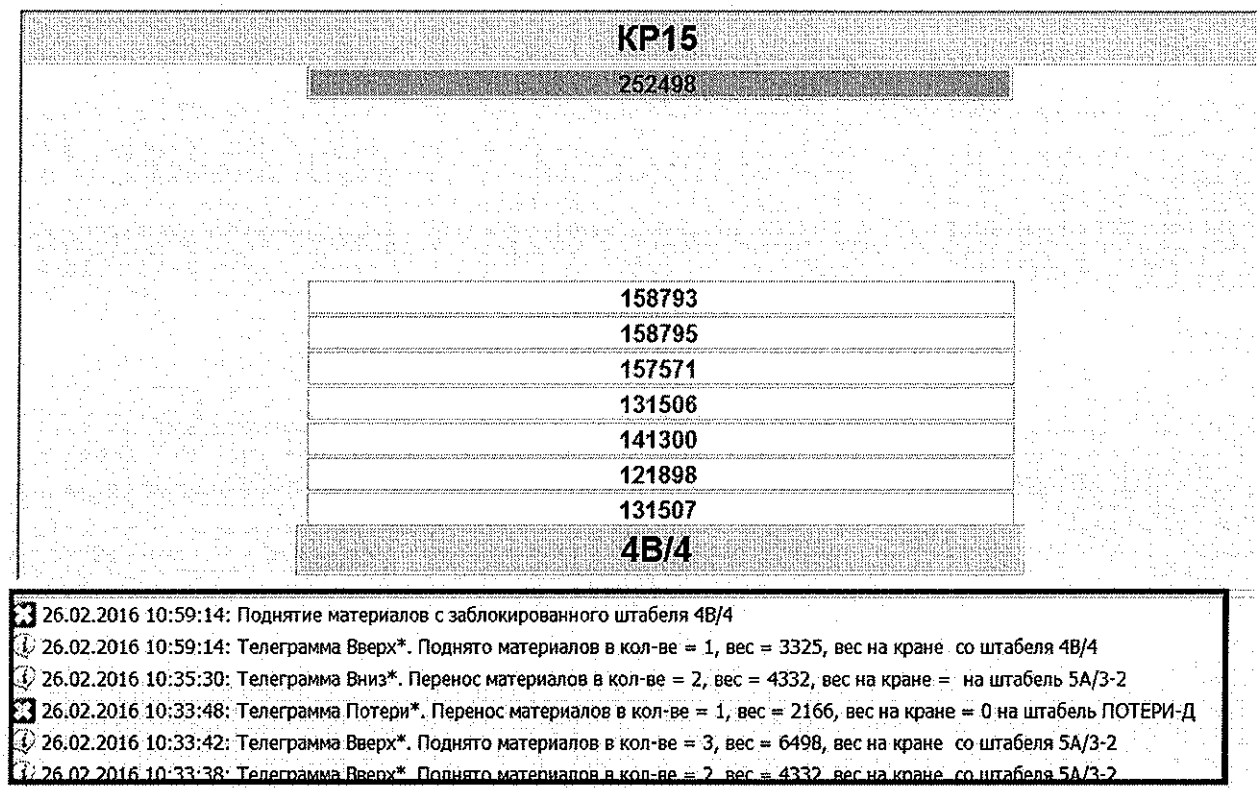
Все эти операции отображаются в списке сообщений.



Следует отметить, что кнопка «Поднять» активна, если кран находится над штабелем, кнопки «Опустить» и «Потеря», когда

8. Сообщения

Каждое событие, связанное с перемещением материала, отображается в списке сообщений, который располагается на вкладке «Графическое отображение» под отображением штабеля. Эти сообщения появляются при перемещении материалов краном по телеграммам, которые он отправляет, а также при корректировке материалов на кране с помощью кнопок «Поднять», «Опустить» и «Потеря».



Сообщения бывают информационные и обозначаются значком  и критические  .

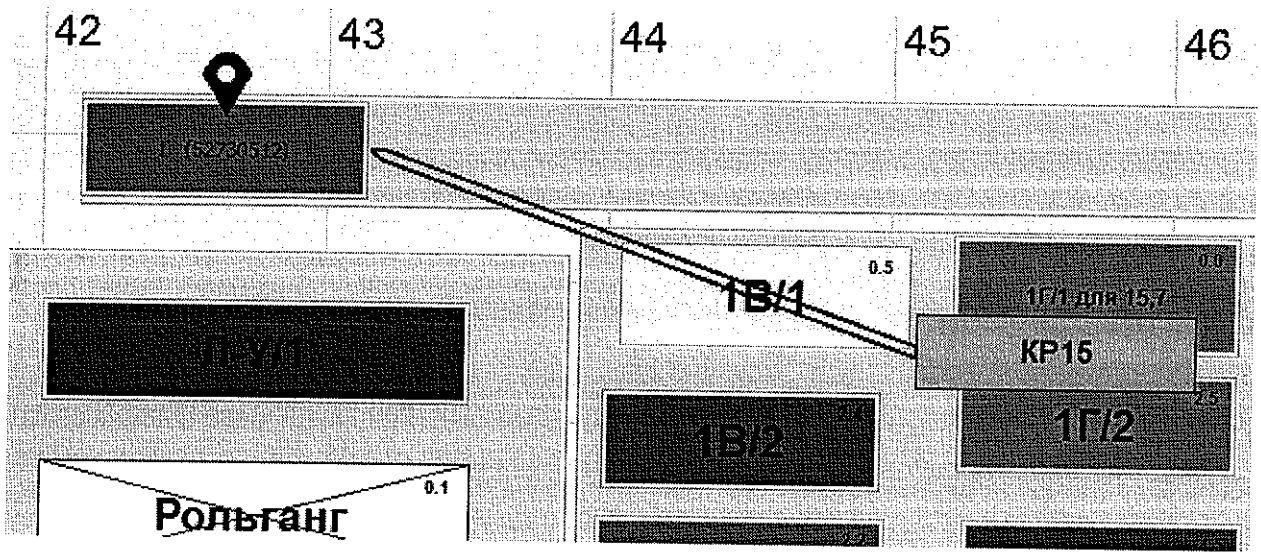
При появлении критического сообщения в списке, оно мигает красным.

9. Калибровка

Для калибровки вагона или штабеля бригадир создаёт отдельное рабочее задание. Оно появляется в таблице рабочих заданий и от других заданий отличается тем, что в нём отображается только Цель – номер вагона.

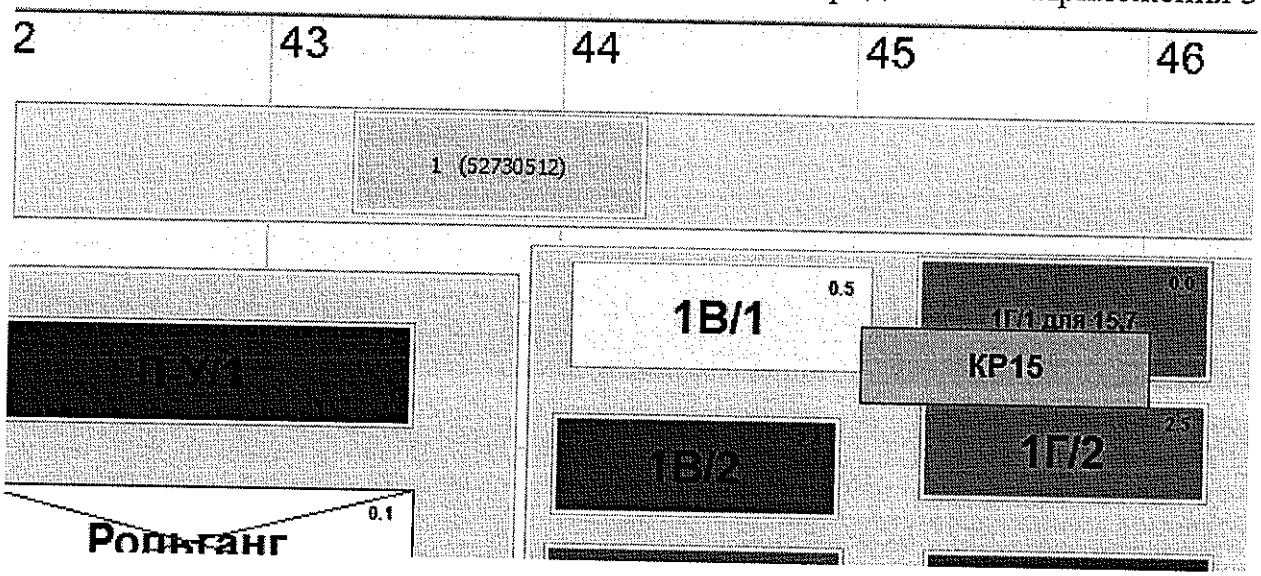
Кол-во	Номер материала	Источник(Спо...	Цель	Вес, кг
0/1	ОАО "Уральская сталь/10	3Г/3-1 (136) (1/М)	3Д/3-1	0/2397

После взятия задания на калибровку в работу появляется подсказка по движению крана. Неоткалиброванный вагон отображается на складе не по реальным координатам, так как до калибровки краном они не известны, и имеет красный цвет.



Когда кран в цехе подъедет к калибруемому вагону, нужно нажать кнопку «Калибровать».

При нажатии на эту кнопку вагону присвоятся координаты и он изменит своё местоположение на обзоре склада, соответствующее реальному положению в цехе, а также изменит свой цвет на жёлтый.


2
43
44
45
46
1 (52730512)
1В/1
0.5
1Г/1 для 15/7
0.0
КР15
1В/2
1Г/2
25
Ролькганг
0.1