ХИМИКОТЕХНОЛОГИЧЕН И МЕТАЛУРГИЧЕН УНИВЕРСИТЕТ

ФАКУЛТЕТ ПО ХИМИЧНО И СИСТЕМНО ИНЖЕНЕРСТВО

КАТЕДРА „АВТОМАТИЗАЦИЯ НА ПРОИЗВОДСТВОТО”

маг. инж. Искра Драгомирова Антонова

А В Т О Р Е Ф Е Р А Т

на дисертация

ОБЕКТНО-ОРИЕНТИРАН ПОДХОД В ИНТЕГРИРАНИТЕ СИСТЕМИ ЗА УПРАВЛЕНИЕ

за присъждане на образователна и научна степен „доктор”

по научната специалност

5.2. Електротехника, електроника и автоматика (Автоматизация на инженерния труд и системи за автоматизирано проектиране)

Научен ръководител: проф. д-р Идилия Бачкова

Научно жури: 1. Проф. дтн. Камен Велев

2. Проф. дтн. Георги Попов
3. Проф. д-р Тодор Нешков
4. Проф. д-р Идилия Бачкова
5. Доц. дтн. Красимира Стоилова

София, 2012

Дистертацията съдържа 170 стр. (формат А4), в които са включени 124 фигури, 4 таблици и 198 литературни източника.

Дисертационният труд е обсъден и приет за защита на заседание на разширения научен съвет на научното звено към катедра „Автоматизация на производството" при Химикотехнологичен и Металургичен университет – София, състоял се на 16.07.2012.

Дисертантката работи като асистент в катедра „Автоматизация на производството" при Химикотехнологичен и Металургичен университет – София, където е била редовен докторант.

Основната част от експерименталните изследвания, включени в дисертационния труд, са проведени в същия Университет и в Университета Мартин-Лутер, Хале-Витенберг, Германия.

Защитата на дисертационният труд ще се състои на 25.10.2012 г., от 14.00 часа в зала 424, етаж 4, сграда „А" на ХТМУ.

Материалите по защитата са на разположение на интересуващите се на интернет сайта на ХТМУ (http://uctm.edu/science) и в одтел „Научни дейности", ст. 406, ет. 4, сгр. А на ХТМУ.

ИЗПОЛЗВАНИ СЪКРАЩЕНИЯ

AOSE - Агентно-Ориентираното Софтуерно Инженерство

BDD-Диаграма за дефиниране на блокове

ECC -Граф за управление на изпълнението

IBD- Диаграма на вътрешни блокове

MAS -Мулти-агентни системи

MAST - Методология за моделиране и анализ на системи за реално време

OO - Обектно-ориентиран

PLC - Програмируем логически контролер

RD - Диаграма на изисквания

SIFB - Функционален блок за услуги

SPT - Планиране, изпълнение и времева спецификация

SUC - Системни случаи на приложение

UML - Унифициран език за моделиране

UCD - Диаграма на случаи на използване

ФБ - Функционален блок

Фигури с номера О-* в автореферата представляват обобщени фигури, включващи група от фигури, включени в дисертационната работа под номерата, представени в кръгче на обобщената фигура. Номерацията на останалите фигури и таблиците съответства на тази в дисертационната работа.

Увод

Бързо променящите се пазарни условия налагат изискването за нов тип производствени системи, които гарантират производството на разнообразни, качествени продукти с по-къс жизнен цикъл, реагират бързо и ефективно на променящите се условия на пазара, предлагат ориентирано към клиента производство и интеграцията му с производството на други фирми.

Тези нови изисквания към производствените системи изправят разработката и поддръжката на системите за автоматизация и управление пред нови предизвикателства, които се усилват допълнително и от широкото навлизане в областта на информационните и комуникационни технологии, изразяващо се в появата на голямо разнообразие от съвременни технически средства и нововъведения. Всичко това води както до нарастване на дела на софтуера в областта на системите за управление, така и до повишаване на неговата сложност и размер. От тук възниква и необходимостта от използване на съвременни подходи и методи на софтуерното инженерство. Сред тях, особено значими по отношение на повишаване на качеството и надеждността, и съкращаване на времето за разработка са моделно управляваните подходи и по-специално разнообразието, което предлагат обектно-ориентираните подходи и унифицирания език за моделиране UML.

В тази връзка целта на дисертационния труд е да изследва и предложи обектно-ориентирани подходи за разработка на системи за управление, които от една страна да покриват всички етапи на жизнения цикъл на разработката, а от друга да дадат положителен отговор на предизвикателствата, пред които е изправена разработката на съвременни системи за управление.

I. Обзор на обектно-ориентираните подходи при разработката на системи за управление

1.1. Изисквания към системите за управление и подходите за тяхната разработка

От анализа на изискванията към системите за управление и към подходите за тяхната разработка, могат да бъдат обобщени следните изисквания към съвременните системи за автоматизация и управление:

- Преминаване към децентрализирана архитектурата, базирана на разделяне на знанията продукт/ресурс;
- Осигуряване на абстрактни, генерализирани и гъвкави взаимодействия;
- Постигане на проактивно, реактивно и самоорганизиращо управление;
- Разработка и имплементация на реконфигурируеми разпределени системи за управление.

Удовлетворяването на тези изисквания е свързано с прилагането на нови подходи за разработка, които осигуряват:

- Надеждна концепция за декомпозиция и модулност;
- Възможност за разширения в случаите на нови продукти, машини и съоръжения;
- Платформено независим модел на обща архитектура;
- Използване на унифицирани, капсулирани и многократно използваеми компоненти;
- Моделиране на архитектурата на системата и нейната функционалност от различни гледни точки, контекст и нива на абстрактност;

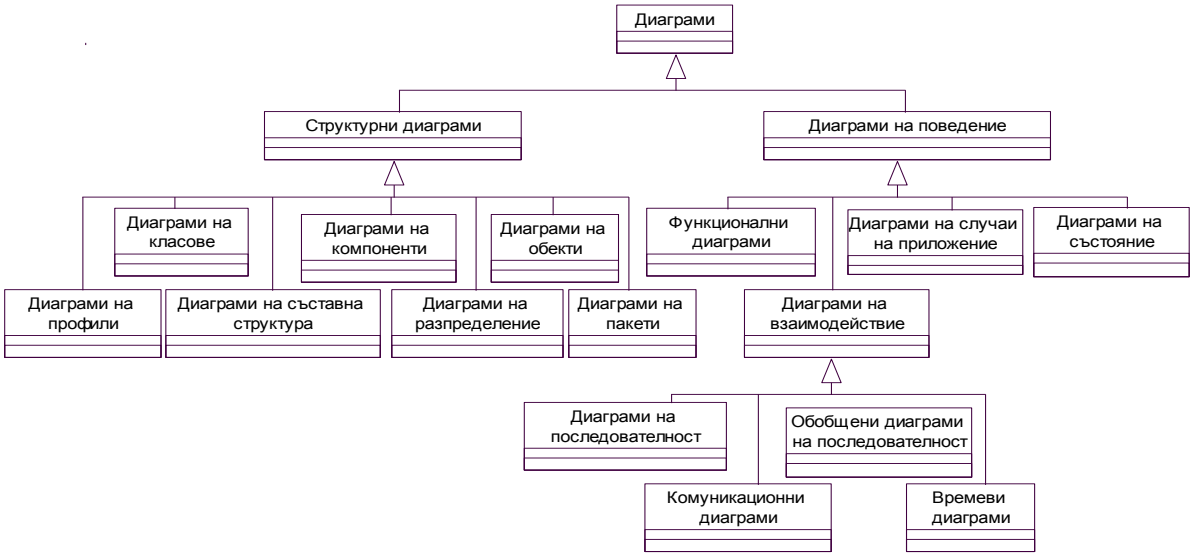
1.2. Кратък обзор на стандарта за моделиране на разпределени системи за управление IEC-61499

Една от основните тенденции, свързана с постигането на разпределеност в системите за управление, се постига чрез използване на стандарта IEC-61499 и в този раздел е направен кратък обзор на стандарта. Той се явява продължение на стандарта за програмируеми логически контролери (PLC) - IEC-61131, като разширява понятието функционален блок (ФБ) със събитийни входове и изходи и граф за управление на изпълнението (ECC), наподобяващ машините на състоянията. Стандартът IEC-61499 дефинира референтна архитектура и модели за разработка на модулни, многократно използваеми, компонентно-базирани и разпределени приложения за управление.

1.3. Основни принципи на обектно-ориентираните подходи

През последните години се наблюдава повишен интерес към областта на обектно-ориентираните (ОО) подходи, намиращи все по-голямо приложение за намаляване сложността на софтуерните системи. Основните характеристики на ОО подходи са капсулиране, наследяване и полиморфизъм, водещи до предимства като модулност, многократна използваемост, разширяемост и оперативна съвместимост.

Кулминацията в развитието на ОО подходи е създаването и използването на UML, който е общоцелеви език за визуално моделиране, представляващ фамилия от графични нотации, обединени в общ мета-модел, използвани за моделиране на различни по вид системи. Чрез UML се моделират различни гледни точки и аспекти на системите, използвайки множество от диаграми и нотации. Диаграмите, използвани в UML, са представени на фиг.1.11 и са разделени на 2 основни групи – структурни диаграми и диаграми на поведение.



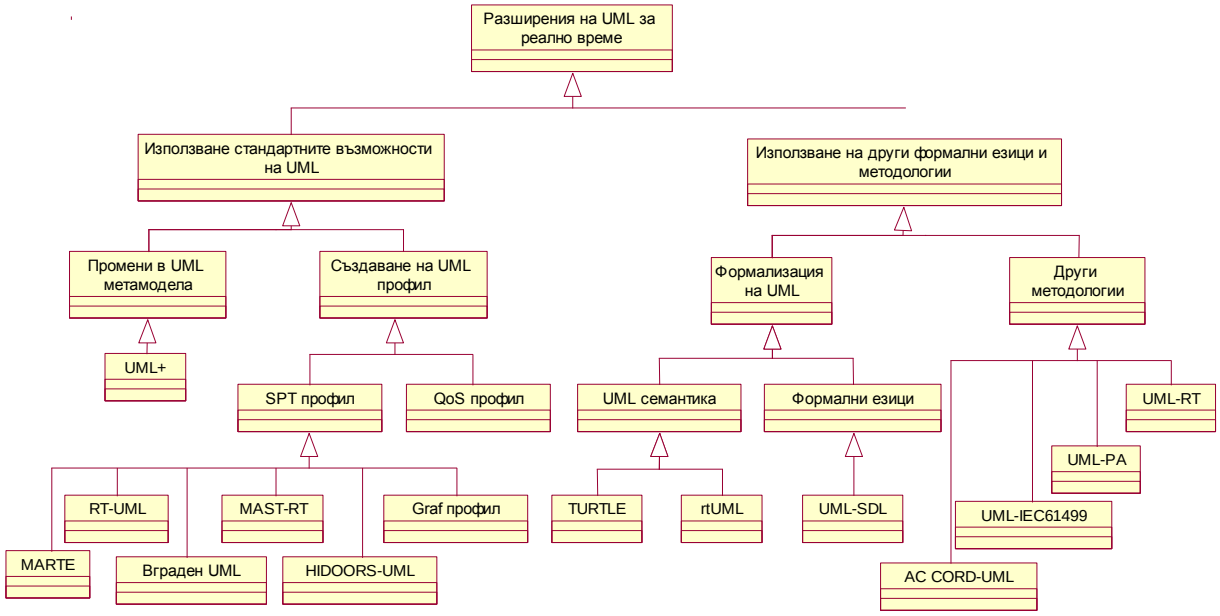
Фиг. 1.11: Класификация на диаграмите в UML

1.4. Концепции за използването на UML за системи за управление

Приложимостта на UML е всеобхватна, но в центъра на анализа е приложението му в областта на автоматизацията и системите за управление. В резултат на анализа подходите, използвани в разглежданата област, са класифицирани в 2 основни клъстера, показани на фиг.1.14. Подходите от първия клъстер използват стандартните възможности на UML. Това е възможно по два начина – чрез промени в мета-модела на UML като се добавят нови мета-класове и мета-конструкции или чрез създаването на профили, базирани на стереотипи, ограничения, или тагови стойности. Едни от първите профили са профила за

условията на планиране, изпълнение и времева спецификация или накратко SPT профил и профила за качеството на услугите (QoS профил). Профилът SPT е използван като основа за създаване на други UML профили, някои от които са RT-UML, MAST-RT, вграден UML, HIDOORS UML профил, Graf-Ober профил и др.

Вторият клъстер от разширения на UML за целите на разработка на системи за управление е свързан с комбиниране на стандартните възможности на UML с възможностите на други рамки, езици и методи за реално време. Тези разширения са обобщени в два основни подкласа, свързани с формализацията на UML и комбинирането му с други методологии. В областта на управлението и автоматизацията, многообещаващ подход, прилаган за разработка на разпределени системи за управление, е съвместното използване на UML и методологията, предложена в стандарта IEC-61499.



Фиг.1.14: Приложение на UML за реално време

1.5. Сравнение на използваните подходи

UML мета-моделът се използва основно за моделиране на софтуерни системи, като са добавени времеви характеристики на данните. Недостатък на мета-модела е, че ключовите му елементи не се поддържат изцяло от стандарта. За разлика от него UML-MAST подходът осигурява стандартни средства за проектиране на приложения в реално време. Въпреки достойнствата, които притежава UML-MAST, не съществуват средства за количествен анализ, поради което се разчита на интеграция с UML профилите за планиране, разработвани от OMG.

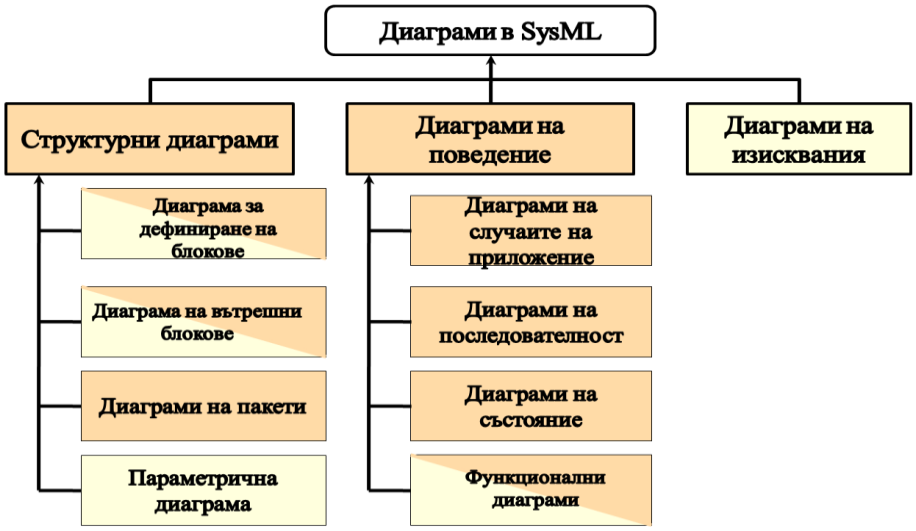
При използването на допълнителни конструкции в средата на UML се използват конструкциите, описващи сложни системи. Недостатък на този подход е, че пълната спецификация на структурата на системи за управление зависи само от комбинацията на диаграмите на класове и сътрудничество.

UML моделът и SDL спецификацята дават ясен поглед върху различните абстрактни нива в системата, като използват правила за преход, дефиниращи промяната на UML модела в SDL спецификация. Недостатък на комбинацията UML-SDL е нееднозначното разпределяне на структура и поведение в модела. Подходът, обединяващ UML с ФБ на IEC-61499, осигурява проектиране и анализ в процеса на разработка, софтуерна капсулация и многократна използваемост на приложението, гарантиращо модулност, разширяемост, оперативна съвместимост и гъвкавост на системата.

1.6. Обзор на профила за системно инженерство SysML

С оглед на осигуряване на възможност за детайлно моделиране на обектите за управление чрез въвеждането на нови нотации и диаграми, специално внимание е посветено на профила за системно инженерство SysML. Профилът осигурява и възможност да се моделира йерархичността на системите, системите да бъдат декомпозирани, както и възможност за дефиниране на изискванията към системите и обектите за управление.

SysML използва подмножество на UML чрез допълнителни разширения от стереотипи, диаграмни разширения и библиотека от модели. На фиг.1.17 обобщено са представени използваните в SysML диаграми и аспектите им на моделиране. Използваното подмножество на UML, известно още като UML4SysML включва диаграми на взаимодействие, машини на състояние, случаи на приложение и профили. SysML преизползва седем от тринадесетте диаграми на UML 2.0 и добавя две нови диаграми – диаграми на изискванията и параметрични диаграми, а също така и таблици на разпределение, които могат да бъдат доставени динамично от диаграмите в UML. Диаграмите на обекти, комуникационните и времевите UML диаграми нямат съответствия в SysML.



Фиг.1.17: Диаграми в SysML

1.7. Кратък обзор на агентно-ориентираните системи за управление

Агентно-базираните системи осигуряват нов начин за анализ, проектиране и имплементация на сложни софтуерни системи. Възможностите за моделиране на мулти-агентни системи на високо ниво на абстракция е тема на изследване от областта на Агентно-Ориентираното Софтуерно Инженерство (AOSE), която се развива много активно през последните години и изследва възможностите на UML и разширенията му за моделиране на основни агентно-ориентирани понятия, като агенти, онтологии и протоколи на взаимодействие. С развитието на UML и въвеждането на активните обекти, разликата между обекти и агенти намалява, благодарение на подобрението на езика на мета-ниво и създаването на нови нотации, както и формализацията на различни диаграми, използвани в езика.

1.8. Изводи към обзора, цели и задачи на дисертационния труд:

От направения литературен обзор могат да бъдат направени следните по-важни изводи:

1. UML намира широко приложение за проектиране и разработка на информационни системи от разнообразно естество и може да бъде приложен за моделиране на софтуерни системи за управление.

2. Задачата за разширяване на възможностите му за проектиране и анализ на системи за управление, продължава да е актуална и перспективна.
3. Подходите и концепциите за моделиране на системи за управление, използвайки UML-RT, частично гарантират основните им характеристики.
4. Понастоящем нито един от подходите и концепциите за UML-RT не позволява гарантирано високо качество при моделиране на системи за управление.
5. Възможностите за детайлно моделиране на физически системи, както и възможността за поддържка на целия жизнен цикъл на разработката на система за управление, започвайки от дефинирането на изискванията до софтуерната имплементация не се поддържат от чистия UML. Тези възможности са осигурени от UML профила за системно инженерство – SysML.
6. До момента не са известни цялостни решения на системи за управление на промишлени системи в реално време, проектирани в среда на UML.
7. При комбинираното използване на UML с други езици или платформи, при преминаването от една в друга платформа има загуба на части от модела.
8. При обектно-ориентираните подходи и UML не съществува стандартен начин за третиране на времето под формата на времеви спецификации, устройства за време, свойства, ограничения, изисквания и др., както по отношение на тяхната функционалност, така и по отношение на техните времеви и качествени показатели.
9. Някои от съществените недостатъци на чистите обектно-ориентирани подходи в задачите за разработка на разпределени системи за управление могат да бъдат преодолени чрез постигането на компонентност на базата на комбинирането им със стандарта IEC-61499.
10. Като перспективно направление за компенсиране недостатъците на обектно-ориентирания подход е използване на концепцията на мулти-агентните системи чрез прилагане принципите на агентно-ориентираното софтуерно инженерство (AOSE).

В резултат на анализа на научните резултати и направените изводи към дисертационния труд е поставена следната **цел**: *да се изследват и предложат обектно-ориентирани подходи за разработка на съвременни системи за управление, гарантиращи съкращаване на жизнения цикъл на разработката, намаляване на разходите, свързано с повишаване степента на многократна използваемост на моделите и компонентите, и повишаване на точността на разработените модели и софтуер*.

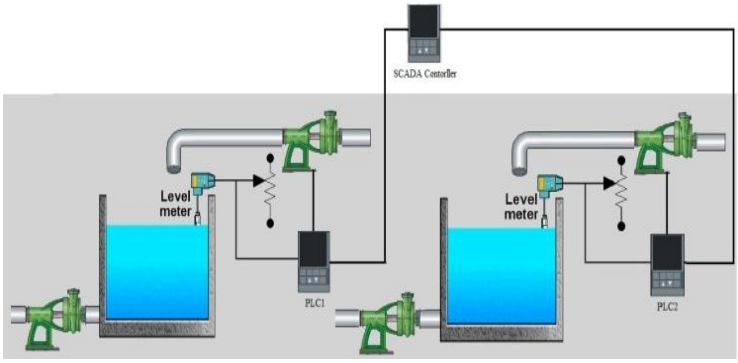
Постигането на поставената цел изисква решаването на следните по-конкретни **задачи**:

1. Да се направи сравнителен анализ на съществуващите обектно-ориентирани подходи за моделиране на системи за управление на базата на избора на представители от клъстерите, дефинирани в Глава първа на дисертацията.
2. Да се изследва приложимостта на тези подходи за моделиране на различни по вид производствени системи: непрекъснати, дискретни.
3. Да се предложи модел на жизнения цикъл (софтуерен процес) за моделно управлявана разработка на системи за управление с използване възможностите на обектно-ориентирания подход и UML.
4. Да се разработи подход за моделно управлявана разработка на системи за управление с възможност за моделиране на затворената система за управление чрез използване на профила за системно инженерство SysML.
5. Да се изследват възможностите за създаване на комбиниран подход за разработка на разпределени системи за управление чрез комбинирането на UML/SysML и стандарта IEC-61499.
6. Да се анализира приложимостта на предложените подходи за моделно управлявана разработка на различни по вид производствени системи (непрекъснати, дискретни, хибридни).

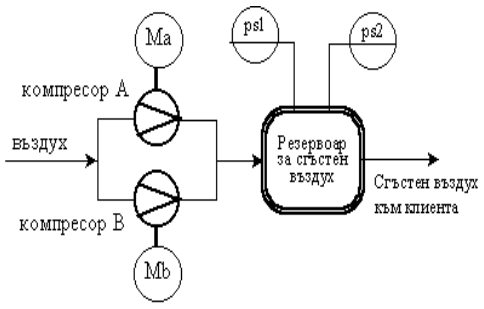
7. Да се изследват възможностите за разширяване на предложения подход към постигане целите на мулти-агентните системи за управление със средствата на обектната ориентация.

II. Сравнителен анализ на подходите за моделиране на системи за управление с UML

Във втора глава е направен сравнителен анализ на подходите за моделиране на системи за управление с UML, като три от подходите, анализирани в първа глава и намерили най-голямо приложение, са приложени за разработка на системи за управление на непрекъснати процеси (фиг.2.1) и система за управление на дискретни процеси (фиг.2.9). Това са: подход от първия клъстер (фиг.1.14), използващ стандартните възможности на UML, подход от същия клъстер, но с използване на допълнителни профили - UML-MAST подхода, както и подход, обединяващ възможностите на UML със стандарта IEC-61499, който е подход от втория клъстер.



Фиг.2.1 Система за управление на ниво в два резервоара



Фиг.2.9: Система въздушни компресори

Таблица 1: Входове и изходи

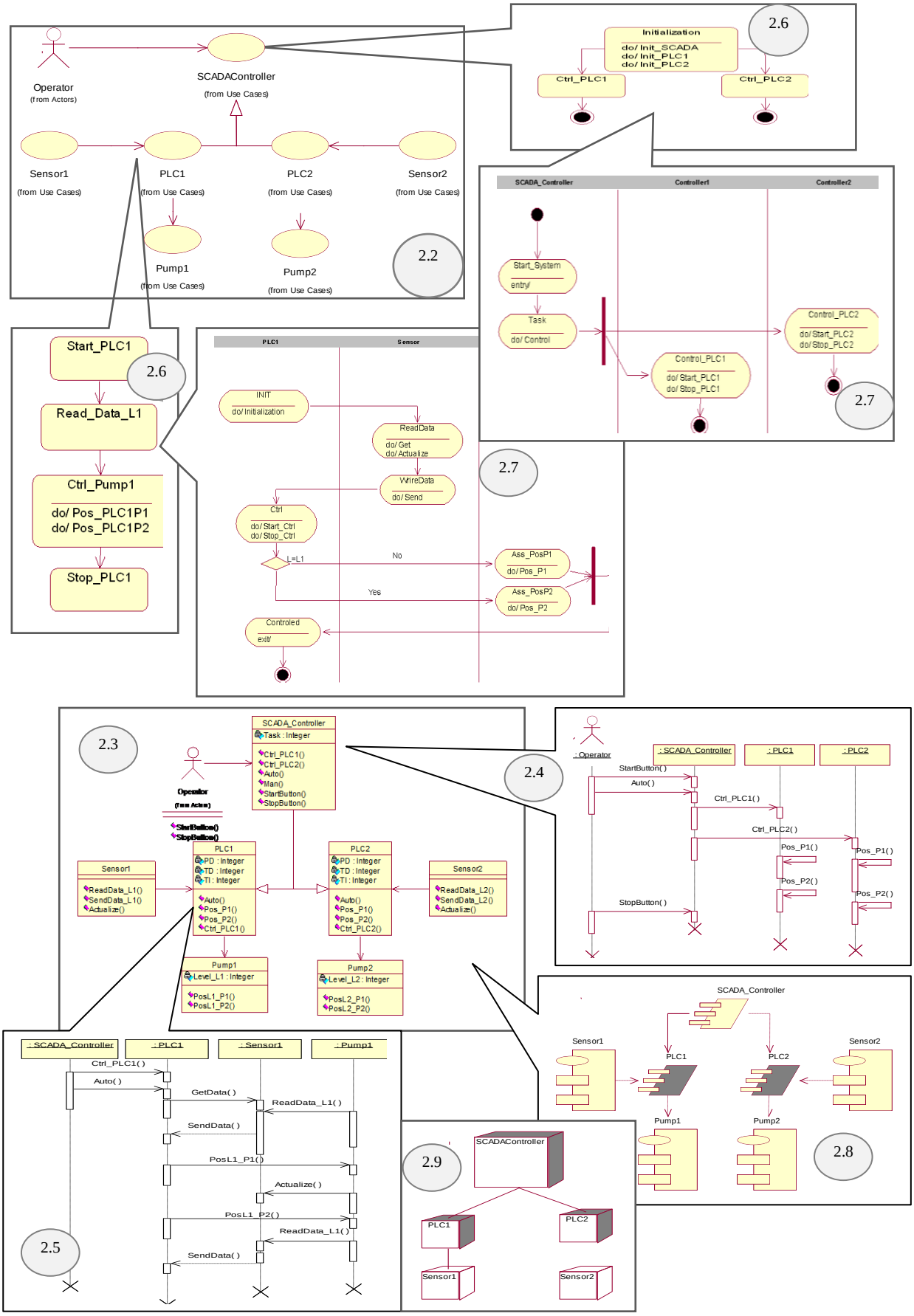
Сензори	Описание
PS-1	Налягането е под 6.1 bars
PS-2	Налягането е под 5.9 bars
Dist_Ma	Мотор A е повреден
Dist_Mb	Мотор B е повреден
Изпълнителен механизъм	Описание
Ma	Мотор A работи
Mb	Мотор B работи

2.1. Подход, използващ стандартен UML 1.4

От версия UML1.4 стартира използването на UML за моделиране на системи за управление. Чрез прилагането на подхода се проверява тезата на Дъглас за възможността с версия UML1.4 да се моделират системи за реално време. Подходът осигурява възможност за моделиране на различни изгледи - случаи на приложения, логически изглед, компонентен изглед и др., използвайки различни диаграми. Подходът не се базира на методология, поради което няма определена последователност и ограничения при създаването на модела.

Чрез диаграмата на случаите на приложение (фиг.О.1-2.2) са представени функционалните изисквания към системата за управление на нивото в два резервоара. Диаграмите на състояние включват информация за различните състояния, в които обектът може да съществува, както и при какви условия обектът преминава от едно състояние в друго и какви действия (алгоритми) се извършват във всяко едно състояние. На фиг.О.1-2.6 са

представени състоянията за SCADA контролера и за контролера PLC1. Действията, извършвани при определен сценарий за SCADA контролера и контролера PLC1 са представени в графичен вид чрез функционалната диаграма на фиг.О.1-2.7.



Фиг.О.1: Илюстрация на подхода използващ стандартен UML 1.4

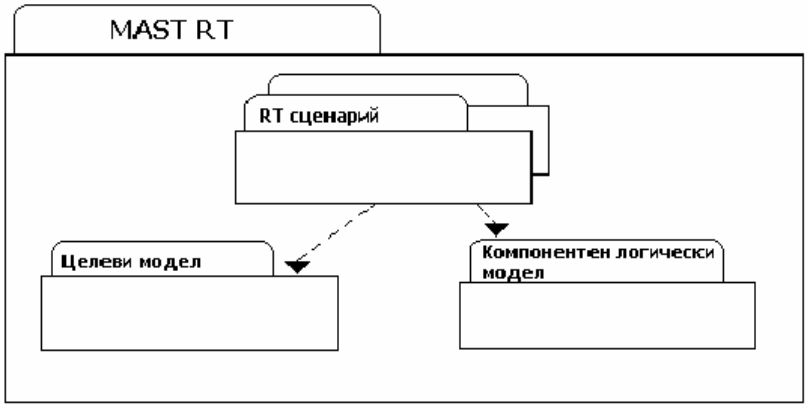
Архитектурата на системата се представя с диаграма на класове (фиг.О.1-2.3), а последователността, в която се изпълняват заявките между SCADA контролера и всеки от контролерите за управление на ниво PLC1 и PLC2, се представя с диаграмата на последователност от фиг.О.1-2.4. Последователността от действия на контролера PLC1 за управление на нивото в резервоара, се моделира с диаграмата на последователност на фиг.О.1-2.5. Архитектурата на системата е представена и с диаграма на компонентите (фиг.О.1-2.8), което позволява автоматично генериране на код след разпределяне на компонентите към класове чрез дефинирани стереотипи. С диаграмата на разпределение (фиг.О.1-2.9) се изобразява как се конфигурира хардуерната част от системата със софтуерните елементи и кои софтуерни елементи от модела към кои хардуерни се разпределят.

За приложение на подхода при моделиране на системата за управление на въздушни компресори са използвани същите по вид диаграми, разработени в същата последователност.

2.2. UML-MAST

В основата на подхода е UML профила MAST-RT, който е илюстриран на фиг.2.19 и включва 3 допълнителни пакета, реализирани като мета-модели:

- Целеви мета-модел (Target_Model) – представя ограничения върху хардуерните и софтуерни ресурси, които съставляват платформата на системата;
- Мета-модел на логически компоненти (Logical_Model) - описва поведението във времето на различни компоненти, което е свързано с удовлетворяване на поредица от изисквания за време;
- Мета-модел на RT сценарии (RT_Scenario) - представят хардуерни/софтуерни режими на работа на системата, за които са дефинирани определени изисквания за реално време.

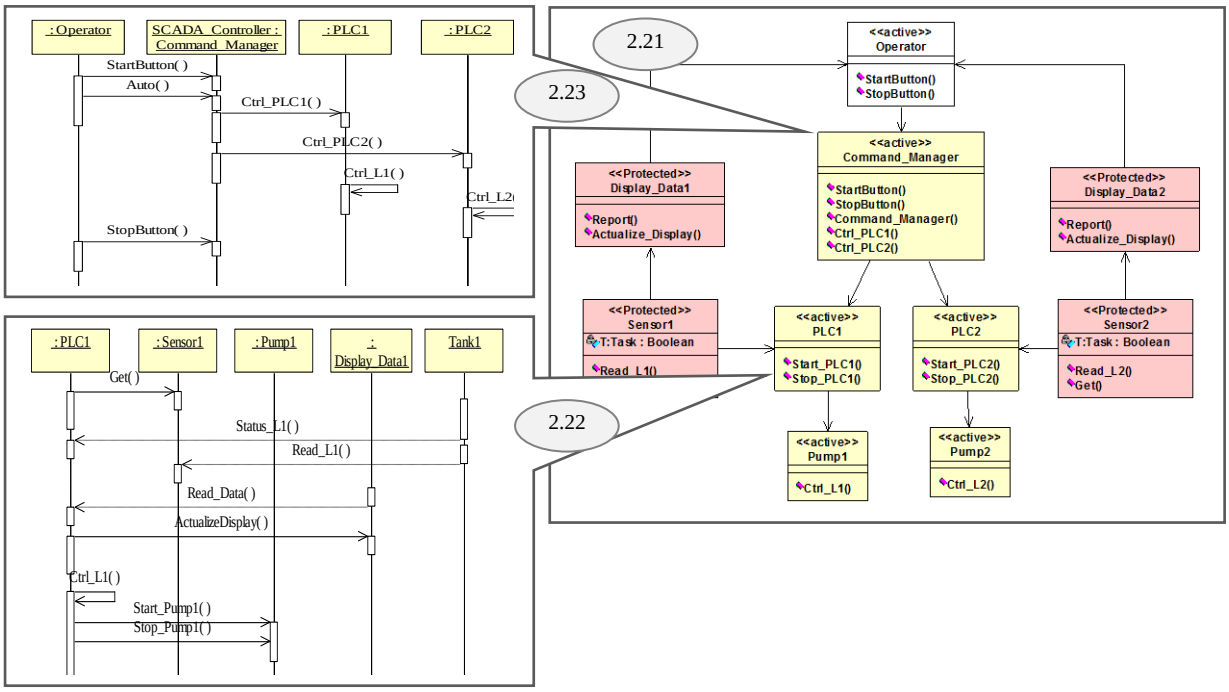


Фиг. 2.19: MAST-RT изглед

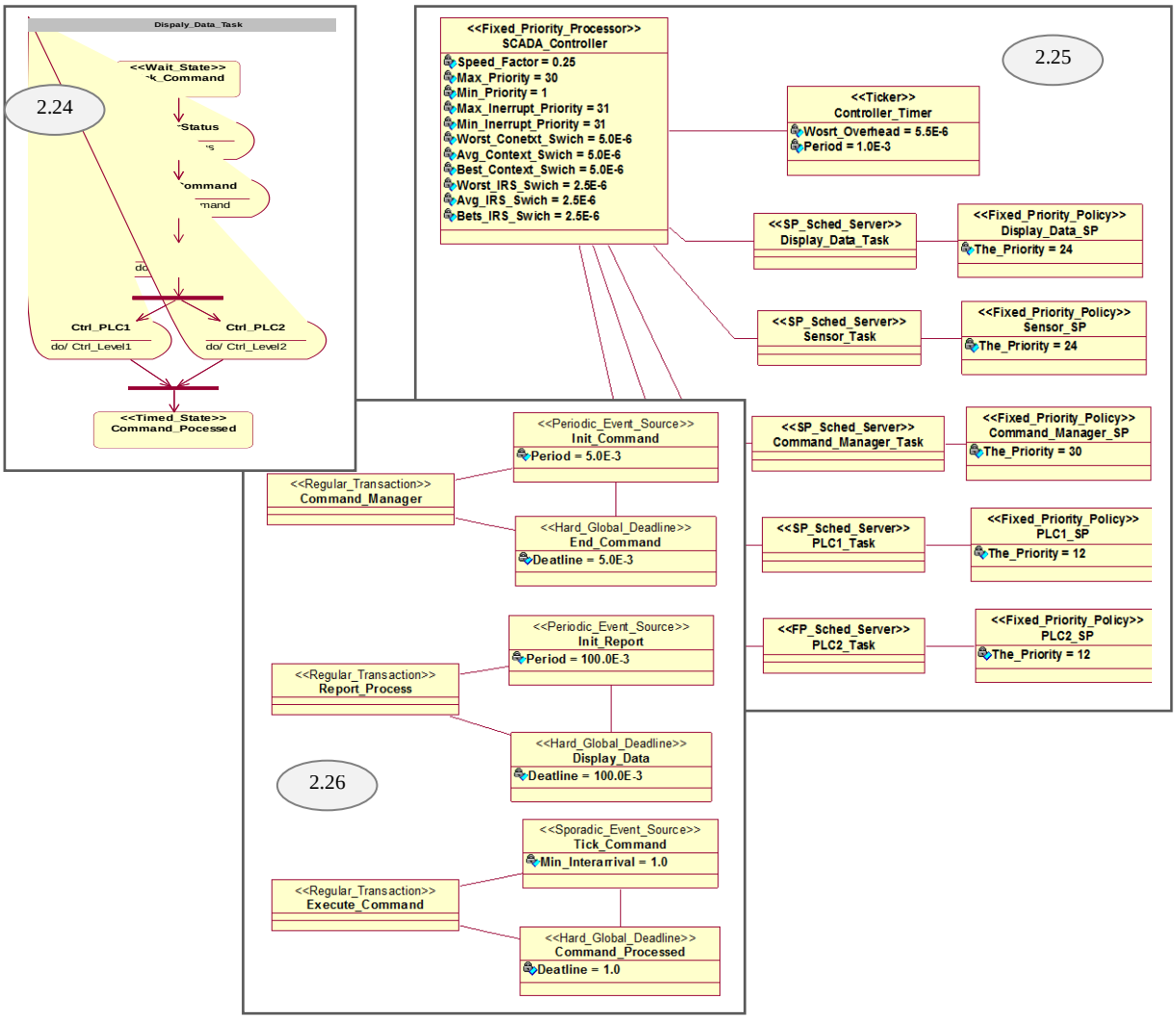
Приложението на подхода за разработка на система за управление на ниво в два резервоара е представено на обобщаващите фиг.О.2 и фиг.О.3. Софтуерната архитектура на системата се представя с диаграмата на класове (фиг.О.2-2.21). Използвани са активни и пасивни класове и връзките между тях. Класовете в диаграмата са екземпляри на UML-MAST мета-класовете. Използвайки диаграмите на последователност (фиг.О.2-2.22 и фиг.О.2-2.23) под формата на съобщения се представят задачите, изпълнявани в класове “Command_Manager” и „PLC1“.

Режимът на работа, представен под формата на RT сценарий е моделиран с функционалната диаграма от фиг.О.3-2.24. Целевият модел, представящ ограниченията на хардуерните и софтуерните ресурси, е показан на фиг.О.3-2.25. На фиг.О.3-2.26 е представен модел за симулация на дефинираните реално-времеви характеристики,

използвайки диаграма на класове. Активирането на преход става след активиране на едно или повече външни събития, представящи времевите ограничения.



Фиг.О.2:Илюстрация на подхода за системата за логическо управление на ниво



Фиг.О.3:UML-MAST профил за системата за логическо управление на ниво

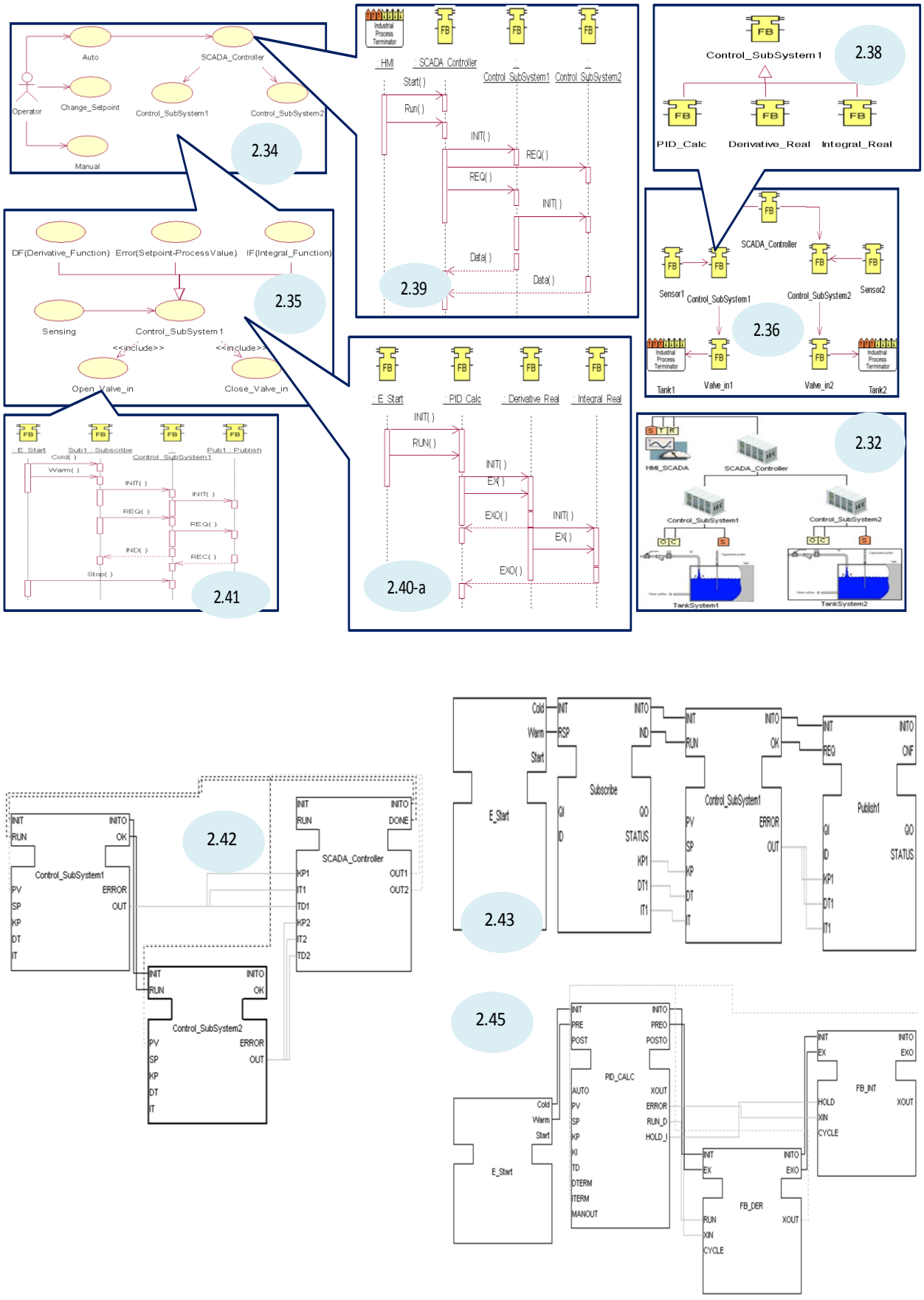
При втория пример, илюстриращ UML-MAST профила и включващ разработката на системата за управление на налягането в резервоар за сгъстен въздух, за етапа, представящ режима на работа, в допълнение са създадени три функционални диаграми, описващи етапа на управление, визуализация и формирането на управляващи въздействия.

2.3. Инженерна среда CORFU – FBDK

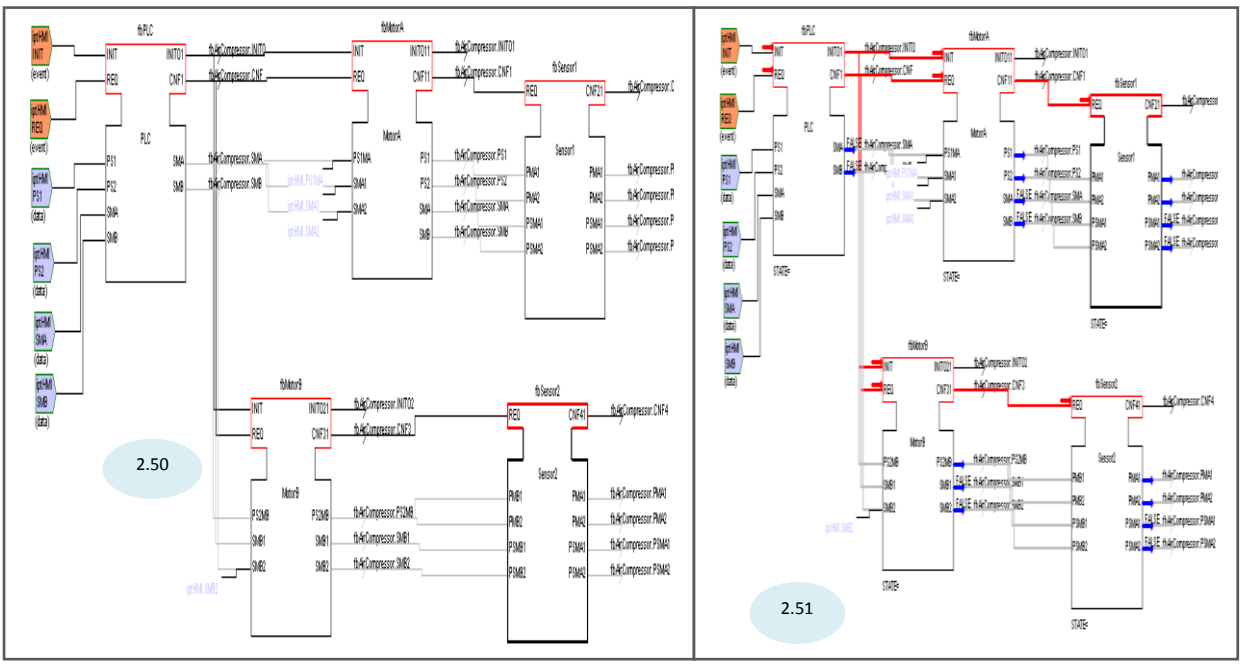
Третият подход, който е анализиран в този раздел е от втория клъстер, представящ възможността за комбинирано използване на UML с други средства. Подходът използва възможностите на UML и стандарта IEC-61499 за моделиране на приложения за реално време. За целта UML е разширен, чрез допълнителните стереотипи - стереотип функционален блок (Function Block - FB), стереотип зависимост по данни (Data Dependency - DD), стереотип зависимост по управление (Control Dependency - CD), индустриален терминатор на процеси (Industrial Process Terminator - IPT) и стереотип реален обект (Real-World Object - RWO). Фазата на проектиране се базира основно на използването на три типа UML диаграми – диаграми на случаите на приложение, диаграми на класове и диаграми на последователност, като връзката със средата FBDK на CORFU се реализира единствено чрез диаграмите на последователност. Архитектурата на CORFU-FBDK поддържа трансформирането на UML модел в модел на функционални блокове, използвайки предварително дефинирани правила за трансформация.

Подходът е приложен към пример за моделиране на система за управление на непрекъснати процеси, обобщен на фиг.О.4. В този случай, като изпълнителни механизми за промяна на нивото в двата резервоара са използвани вентили. Изискванията към системата за управление са дефинирани чрез диаграмите на случаите на приложение (фиг.О.4-2.34 и О.4-2.35), като диаграмата на случаите на приложение от фиг.О.4-2.35 представя разглеждания сценарий за управление, състоящ се в следното: подсистемата за управление получава данни от сензорите и изпълнява действията „Open_Valve_in” или „Open_Valve_out”. Сценариите в случаите на приложение с имена „SCADA_Controller“, „Control_SubSystem1“ и „Open_Valve_in“ са представени с диаграмите на последователност от фиг.О.4-2.39, фиг.О.4-2.40-а и фиг.О.4-2.41. Структурата на приложението е представено чрез диаграми на класове. На фиг.О.4-2.36 е представена диаграмата на класове на горно йерархично ниво. На класовете от диаграмата са присвоени стереотипи „Function block” и „Industrial Process Terminator”, отразяващи аспектите на стандарта IEC-61499. Структурата на подсистемата за управление на ниво е представена с диаграма на класовете от фиг.О.4-2.38. Класът „Control_SubSystem“ е съставен клас и включва в себе си останалите класове, което е отразено с връзка тип „Generalization”. В средата на CORFU-FBDK автоматично се генерира и архитектурата на системата, показана на фиг.О.4-2.32. От създадените диаграми на последователност, автоматично са получени диаграмите на функционални блокове, показани на фиг.О.4-2.42, фиг.О.4-2.43. и фиг.О.4-2.45.

Разработено е и приложение за управление на система за управлението на налягането в резервоар за сгъстен въздух. Структурата на системата, дефинираните функционални изисквания и комуникацията между обектите, изграждащи системата за управление на системата въздушни компресорисе дефинират, както и при системата за управление на непрекъснати процеси. Разликата е, че при системата за управлението на налягането в резервоара, след автоматичното получаване на диаграмата на ФБ (фиг.О.5-2.50) от диаграмата на последователност е извършена валицация на модела чрез симулация. Симулацията може да бъде извършена по отношение на входно-изходните събития, входно-изходните данни и входно-изходните данни и събития. За разработеното приложение е направена симулация по отношение на входно-изходните данни и събития, която е представена на фиг.О.5-2.51.



Фиг.О.4:Приложение на подхода CORFU-FBDKза моделиране на система за логическо управление



Фиг.О.5:Валидация на модела на система от въздушни компресори

2.4. Изводи и заключения към втора глава

От направения анализ на изследваните подходи и разработените модели могат да бъдат направени следните изводи:

1. Изследваните подходи са приложими за моделиране, както на системи за управление на непрекъснати процеси, така и на дискретни.
2. Използването на UML1.4 значително опростява проектирането на системи за управление:
   - моделите са съставени от многократно използваеми, стандартизирани компоненти, съкращаващи времето за разработка на системата, повишавайки надеждността на системите за управление, които дават възможност за създаване на библиотеки за многократното им използване при разработката на нови проекти;
   - използвайки различни видове диаграми, моделите са представени от различни гледни точки;
   - елементите в системата се капсулират, като по този начин се намаляват грешките при проектирането.
3. Недостатък на подхода, базиран на UML1.4 и на използвания софтуер, е невъзможността за представяне на паралелно протичащи процеси в диаграмата на последователност, както и, че с този подход не могат да бъдат моделирани реално времевите аспекти на системата.
4. Методологията MAST-RT дава допълнителни средства за анализ на модела в реално време. Основна отличителна характеристика на разглежданата методология е, че тя позволява моделиране на различни хардуерни аспекти на системата, използвайки независими компоненти за моделиране на системите за реално време, като: логически операции, изпълнение на заявки за реално време, хардуерна платформа и детайли на симулационната система.
5. Недостатък на MAST-RT подхода е липсата на средства за верификация и възможността за генериране на код единствено на езика Ada, което ограничава избора на контролер за управление.
6. Приложен е обектно-ориентиран подход за поетапно проектиране на отворени разпределени системи за управление на базата на стандарта IEC-61499, разширяващ съществуващите възможности на базираните на стандарта средства

чрез тяхното интегриране с моделиращите средства на UML. Това води до по-детайлно и пълно отразяване на изискванията към системата за управление, на нейната структура, поведение и физическа реализация.

7. Подходът дава възможност за капсулиране на елементите на системата, чрез което се намаляват грешките, позволява изграждането на стандартизирани комуникационни канали и се постига по-висока степен на автономност.
8. Моделите са съставени от многократно използваеми, стандартизирани компоненти, които съкращават времето за разработка на системата, повишават надеждността на системите за управление, дават възможност за създаване на библиотеки за разработка на нови проекти и значително опростяват проектирането на разпределени системи.
9. Моделите, разработени с помощта на CORFU-FBDK, имат съществени недостатъци, като:
   - използват се само три типа UML диаграми;
   - създават се само интерфейса на системата и графа за управлениена изпълнение то, а изпълнимите алгоритми трябва да се създават ръчно;
   - симулацията като средство за валидация на моделите не дава задоволителни резултати.

В следващата глава на дисертацията е предложен и представен подход за разработка на системи за управление с използване на UML профила за системно инженерство SysML, който цели да преодолее недостатъците на подходите, разгледани в тази глава.

III. Разработка на системи за управление с използване на UML профила за системно инженерство – SysML

3.1. Представяне на подхода

Една от най-ефективно работещите комбинации от подходи за моделиране на системи за управление е базирана на стандарта UML и профила му за системно инженерство SysML. Предимствата на профила са свързани с моделиране на изискванията към системите за управление и възможностите за комуникация между капсулирани компоненти с висока степен на автономност и са в основата на предлагания в тази глава обектно-ориентиран подход за разработка на системи управление. Предложеният нов подход, използващ възможностите на UML профила за системно инженерство SysML за разработка на системи за управление, е илюстриран на фиг.3.1. Жизненият цикъл на разработката включва следните основни етапи: анализ на изискванията; функционален анализ на система; проектиране на архитектурата; и хардуерно/софтуерно специфициране на проекта.

Фиг. 3.1: Илюстрация на подхода

Фазата на анализ и дефиниране на изискванията започва с анализ на входовете и изходите на системата. Изискванията на клиентите се превръщат в набор от изисквания, като за целта се използват диаграми на изискванията (RD) и случаите на приложение се структурират йерархично и се групират в системни диаграми на случаите на приложение (SUC).

Във фазата на функционалния анализ установените функционални изисквания се превръщат в съгласувани системни функции. Диаграмата на случаите на приложение (UCD) представя функционалната фаза на системата. За всеки случай на приложение се извършва анализ. За представяне на структурите на сложни системи се използват диаграми за дефиниране на блокове (BDD диаграми) и диаграми на вътрешни блокове (IBD диаграми). За моделиране на системната функционалност на високо ниво се използват различни функционални диаграми (BB-AD) и диаграми на последователност (BB-SD) от тип „черна кутия".

Проектирането на архитектурата е третата фаза на предложения модел и включва проектиране на структурата и поведението на подсистемите, основаващо се на варианти на SysML диаграми, използвани в предишната фаза, но от тип „бяла кутия" (WB-UCD, WB-AD, WB-SD). Други важни задачи са свързани с дефинирането на портове и интерфейси и моделиране на поведението на подсистемите посредством SysML диаграми на състояние (SCD). Последният етап е хардуерна/софтуерна спецификация на проекта и е тясно свързан с имплементацията и внедряването на системата. Всички етапи на методологията включват задачи по верификация и валидация, въз основа на моделите, разработени в предходните етапи и на вече дефинираните изисквания.

3.2. Приложение на подхода при разработката на разпределени системи за управление

Предложеният подход е илюстриран с разработката на разпределена система за управление на станция за разпределяне на детайли (фиг.3.2), която е част от пилотната инсталация на корпорацията FESTO, разположена в Университета Мартин Лутер – Хале – Витенберг, Германия. Сензорите и изпълнителните механизми към тези модули са представени в Таблица 3 и Таблица 4. Станцията се състои от два модула - „Storage Control" и „Transfer Control", които се управляват от два отделни контролера, свързани помежду си посредством входно/изходни интерфейси.

Фиг.3.2: Изглед на FESTO станция за разпределяне на детайли

Таблица 3: Сензори и изпълнителни механизми на модула “Storage Control”

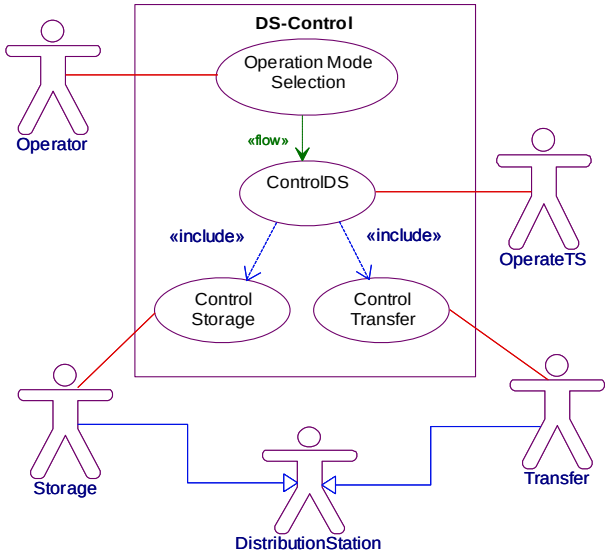
Сензори	
Pos_R	Пушерът е прибран
Pos_E	Пушерът е изтеглен
Empty	Празен магазин
Изпълнителни механизми	
Extend	Изтегляне на пушера

Таблица 4: Сензори и изпълнителни механизми на модула „Transfer Control”

Сензори	
at_Mag	ПУ в позиция “магазин”
at_TS	ПУ в позиция ”съседна станция”
Vac_on	Засмукан детайл
Изпълнителни механизми	
to_Mag	Задвижване на ПУ към позиция “магазин”
to_TS	Задвижване на ПУ към позиция “съседна станция”
VCM_on	Включване на вакуум
VCM_off	Изключване на вакуум

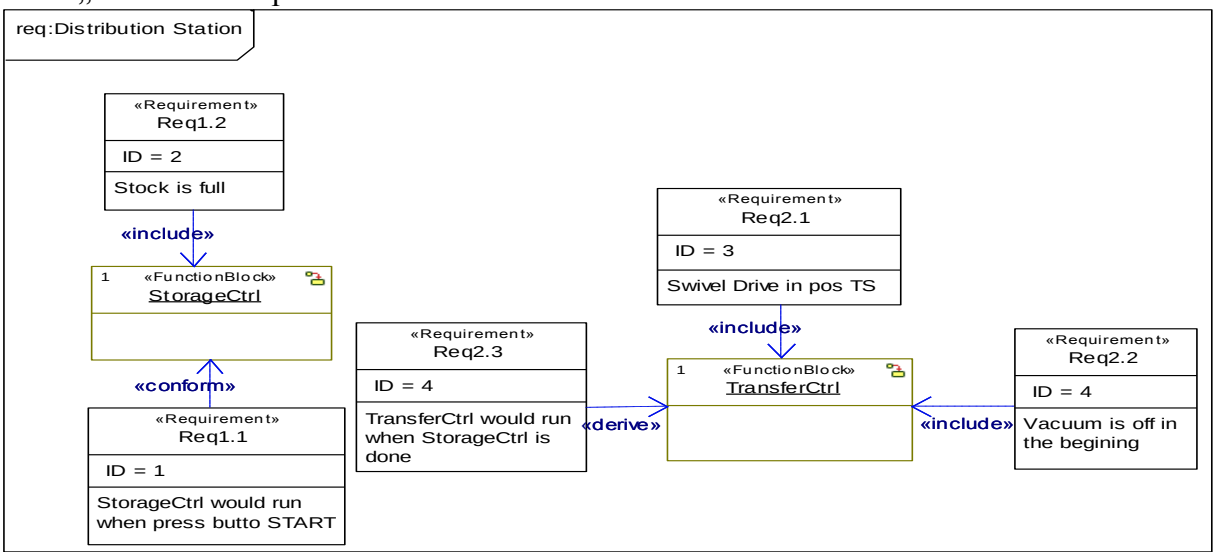
Диаграмата на случаите на приложение (фиг.3.3) е използвана, за да се моделира взаимодействието между системата и нейните потребители, както и изискванията към приложението. Разглежданото приложение може да работи в различни режими като: “старт”, “стоп”, “повторно включване”, “аварийно спиране”, ръчен или автоматичен режим, инициализирани от оператора посредством конзола със съответните бутони, които изискват различна последователност от управляващи въздействия.

Диаграмата на изискванията, представена на фиг.3.4, се използва за създаване на таксономия на предявените изисквания към системата за управление. Изискванията могат да бъдат зададени в текстов или графичен вид. За конкретния пример, необходимите изисквания са дефинирани в графичен вид със средства от булевата алгебра, като: “Магазинът е запълнен с детайли”, т.е. Empty=0 (лъжа), “Цилиндърът е избутан напред”, т.е. Pos_E=1 (истина) и т.н.

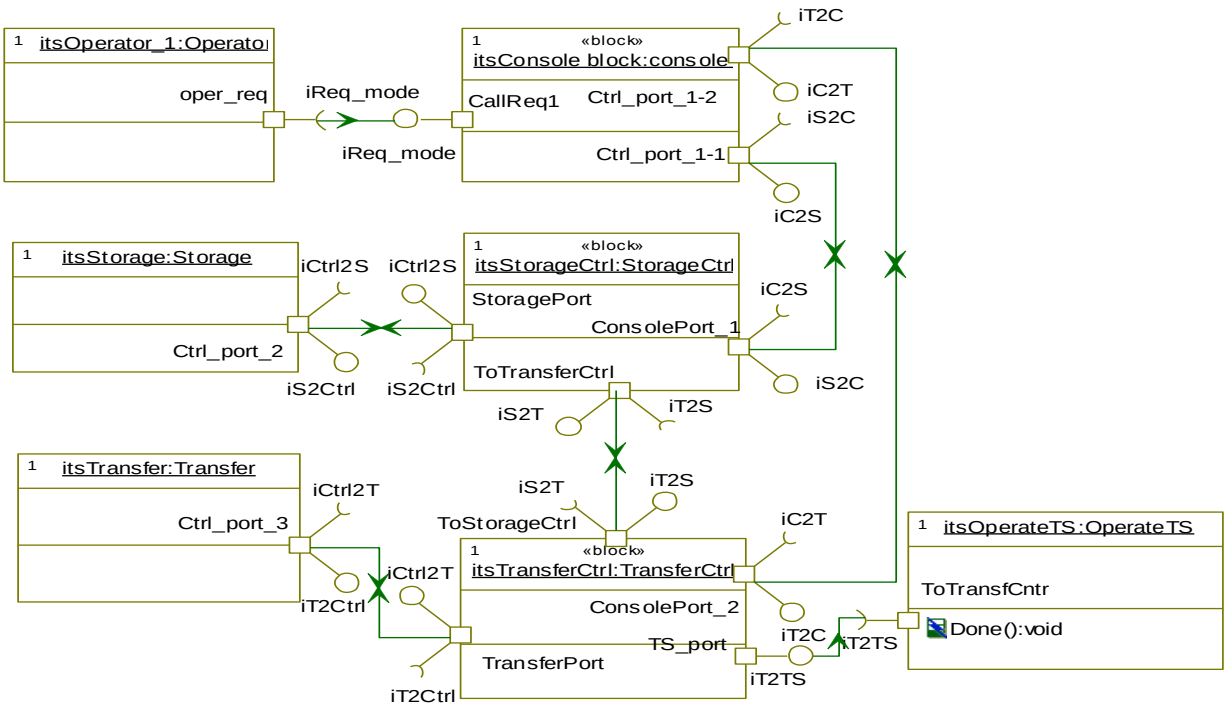


Фиг.3.3: Диаграма на случаите на приложение при управление на FESTO станция за разпределяне на детайли

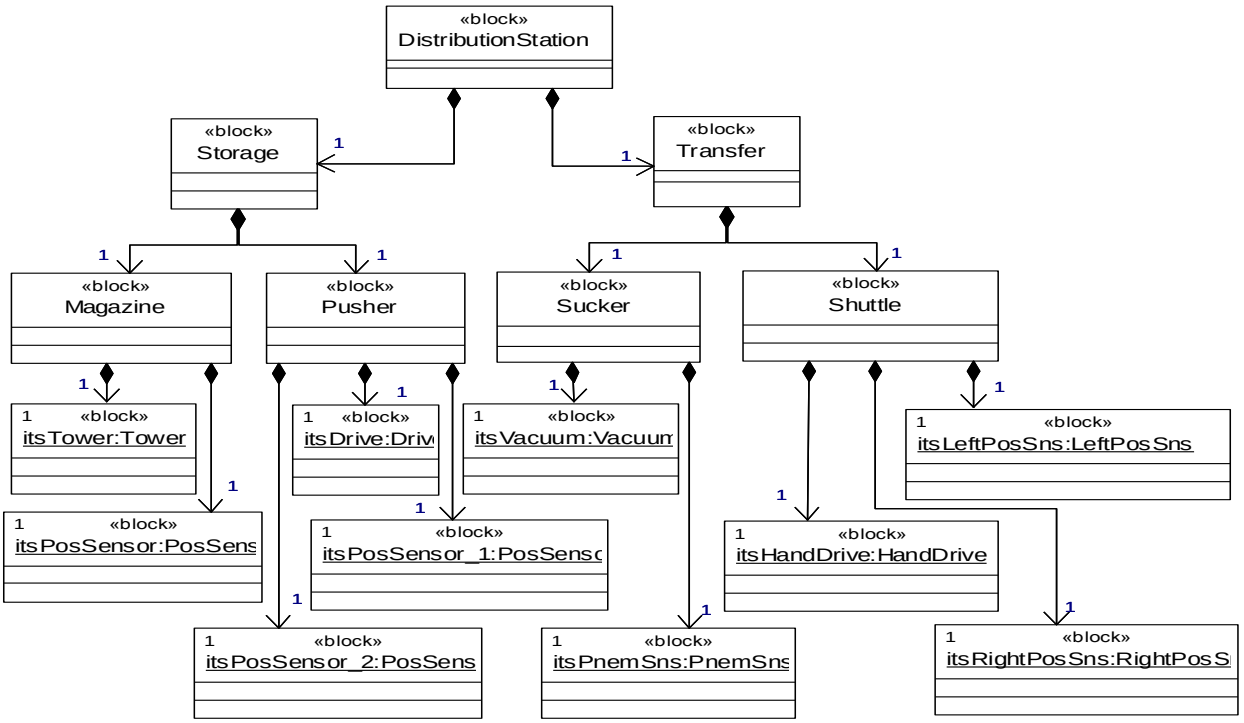
Статичната структура на системата, състояща се от два контролера, конзола, шина тип Profibus и самата станция за разпределяне на детайли, е моделирана чрез BDD, представена на фиг.3.5. Основните елементи на диаграмата за дефиниране на блокове са „системните блокове”, а системните блокове, представящи станцията за разпределяне на детайли, са с имена Console, Operator, Controller, Storage, StorageCtrl, Transfer, TransferCtrl, DistributionStation и OperateTS. Комуникацията между тях се осъществява чрез различен тип портове. Портовете, използвани в приложението, са от тип „provide” и „require”. Използването на комуникационна шина от тип Profibus налага портове от тип „flow” и стандартни портове. Йерархичната структура на станцията за разпределяне на детайли е представена с BDD диаграма (фиг.3.6), състояща се от системни блокове с имена DistributionStation, Storage, Transfer, Magazine, Pusher, Sucker, Shuttle, Tower, PosSensor, Drive, PosSensor_1, PosSensor_2, Vacuum, HandDrive, PnemSns, LeftPosSns и RightPosSns. За да се отрази йерархичната структура, са използвани връзки между системните блокове от тип „Directed Composition”.



Фиг.3.4: Диаграма на изисквания към управлението на FESTO станция за разпределяне на детайли

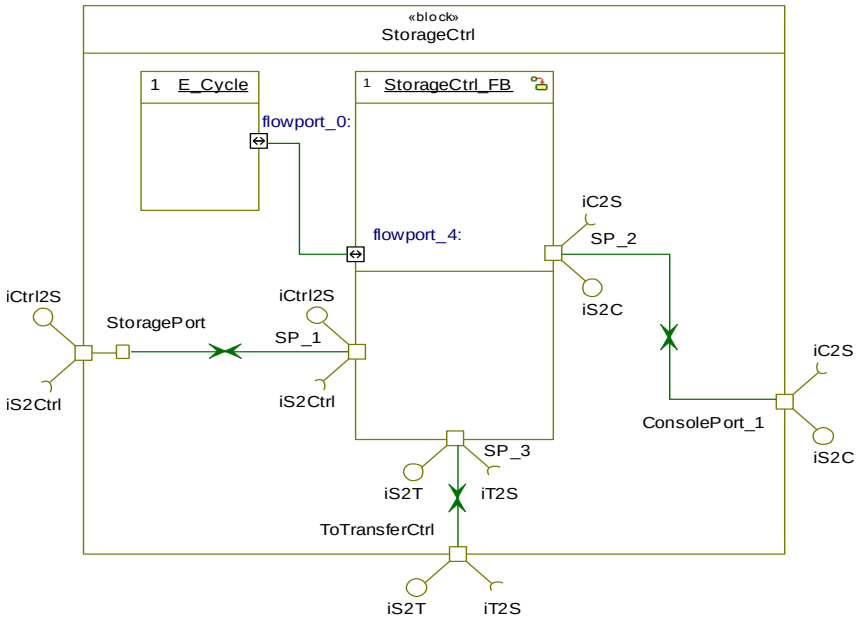


Фиг.3.5: Диаграма за дефиниране на блокове (BDD)



Фиг.3.6: Диаграма за дефиниране на блокове (BDD) на физическата система

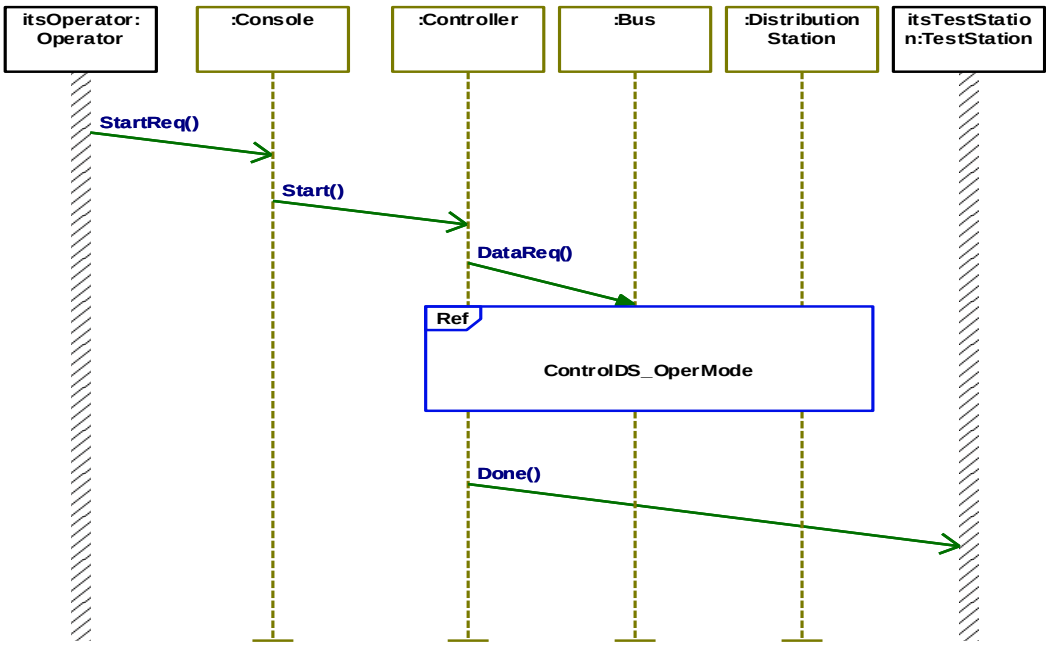
Вътрешната структура на двата разпределени контролера е моделирана чрез IBD диаграма, използвайки основните й елементи - частите (parts) и връзките между тях. На фиг.3.7 е представена вътрешната структура на модула „Storage Control”. Системните блокове са описани с атрибути и операции, които не са визуализирани на фигурата. Атрибутите дават допълнителна информация за състоянията на модулите, а операциите представят действията, извършвани от тях, като: инициализация, старт, стоп, избутване на детайл, включване и изключване на вакуума, преместване на въртящото рамо от една позиция в друга и др. Комуникацията между двата модула се реализира с порт тип „flow”, а комуникацията между модулите и подсистемата за управление чрез стандартен тип портове, които са специфицирани за дефинирания интерфейс чрез стереотипите „provide” и „require”.



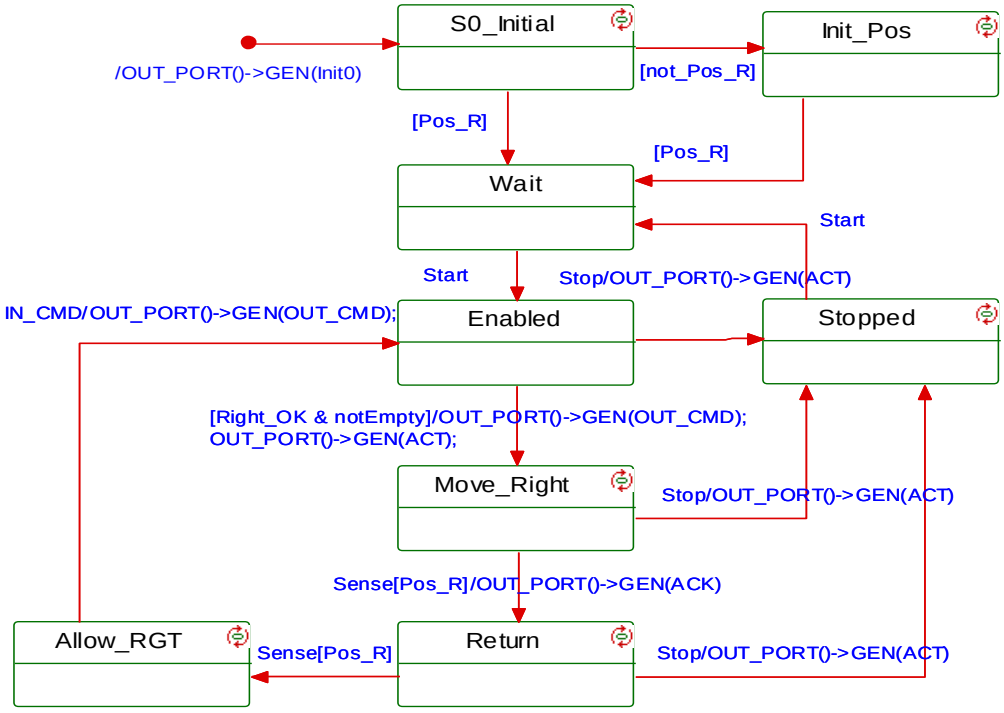
Фиг.3.7: Диаграма на вътрешни блокове (IBD)за модула „Storage Control”

Динамичното поведение на системата и нейните части са моделирани с Диаграми на последователност, Диаграми на състояние и Функционални диаграми. Съобщенията, обменяни между системните блокове, са показани чрез диаграмата на последователност от фиг.3.9.

Друг изглед от динамичното поведение на системата се представя с диаграма на състояние за модула „Storage Control" показанана фиг.3.10. Диаграмата описва неговите основни състояния и връзките при преминаване от едно състояние в друго чрез така наречените преходи.



Фиг.3.9: Диаграма на последователност за нормален режим на работа на станцията за разпределяне на данни



Фиг.3.10: Диаграма на състояние за модула „Storage Control"

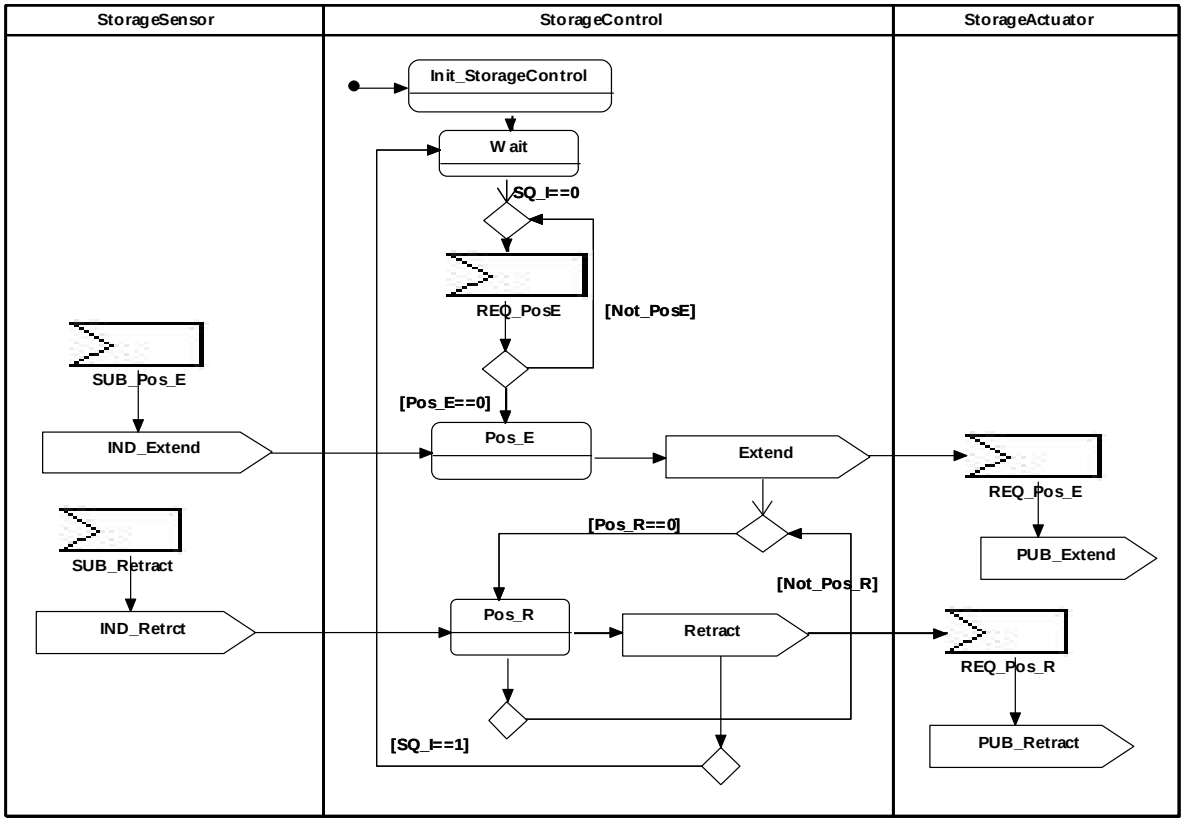
От начално състояние обектът преминава в състояние на инициализация „S0-Inital". В това състояние се инициализира паметта на контролера, като на изходните данни

“Allow_Right” и “Allow_Left” се присвоява стойност “false”. От това състояние обектът (управлението) може да премине в две нови състояния „Init_Pos” и „Wait”. В случай, че пушерът е в позиция „назад”, т.е. е в сила условието [pos_R], управлението преминава в състояние „Wait”, в което не се извършват никакви действия. В противен случай, т.е. ако е изпълнено условието [not pos_R], обектът „StorageCtrl” преминава в състояние „Init_Pos”. В него се инициира действие за връщане на пушера в позиция „Retract”, като на изходната данна retract се присвоява стойност “true” (retract:=1) и се генерира изходното събитие „ACT”, („ACT/ pos_R”) OUT_PORT(cc_mm)->GEN(ACT), свързано с привеждане на изпълнителния механизъм в действие по връщане на пушера в позиция „Retract”.

При удовлетворяване на условието [pos_R], т.е., когато пушерът е вече в позиция „Retract”, управлението преминава в състояние на изчакване „Wait”. От състояние на изчакване обектът може да бъде изваден при настъпване на събитието „Start”, при което се преминава в състояние „Enable”, в което обектът попада и от състояние „Allow_RGT”, при възникване на събитие “IN_CMD”. Обектът може да бъде изведен от състояние „Enable” по два начина: или при появата на събитие “Stop”, при което минава в състояние “Stopped”, или при удовлетворяване на условието за прехода [RIGHT_OK & notEMPTY] (което означава, че детайлът може да бъде поет от транспортьора и магазинът на склада не е празен), когато се преминава в следващото състояние „Move_Rgt”. В това състояние, на изходната данна “Retract” се присвоява стойност “false”, а на “Allow_Right“ – стойност “true” като се генерират изходните събития “ACT” и “OUT_CMD”, свързани респективно с посочените данни. От това състояние при настъпване на събитие “Stop” се преминава в познатото вече състояние “Stopped”, а при настъпване на събитие „Sense” и условие на прехода [pos_E] (пушерът е изтеглен), обектът преминава в състояние на връщане на пушера „Return”, в което на изходната данна “Retract” се присвоява стойност “true” и се генерира изходно събитие “ACT”. Както в предишните състояния, така и тук при възникване на събитието “Stop” се преминава в състояние “Stopped”, а при възникване на събитие „Sense” и условие на прехода [pos_E] (пушерът е изтеглен), обектът преминава в състояние „Allow_Right”, в което на променливата “Allow_Right” се присвоява стойност “true” след което се генерира изходното събитие “OUT_CMD”. При възникване на събитието “IN_CMD”, обектът се връща отново в състояние „ Enabled”. Изходните събития, които се генерират са „ACT/ stop” и „ACT/reset”. Разгледан е нормалният режим на работа, в който след инициализация и стартиране, се преминава към състояние „Extend“ ако е изпълнено условието [not_empty]. Ако условието не е изпълнено, модулът остава в състояние „Wait”, а при изпълние на условието в състояние „Extend“ се изпълнява прост алгоритъм за преместване на пушера, представен с булевия израз extend:=1; retract:=0. След изпълнението на алгоритъма се генерира изходното събитие ACK/Extend=true (потвърждение на избутването). Изпълнението на това събитие се използва като активиращо условие на следващото състояние с име „Retract”, в което се изпълняват алгоритми за връщане на пушера в изходна позиция. Алгоритмите са представени със следните изрази: allow_right:=1;Retract:=1;Extend:=0. Алготитъмът allow_right:=1 се използва като гранично условие, с което се осъществява комуникацията със съседния модул (управление на трансфера). Изпълнението на алгоритмите в това състояние води до генериране на изходното събитие ACK/ Retract =true (потвърждение на издърпването). Това се изполва като активиращо събитие (Sense[Pos_R=1]) за преминаване към следващото състояние – „Allow_Right“, което се използва за генериране на събития за комуникация с модула за управление на трансфера. Изходното събитие OUT_CMD за модула на склада генерира входното събитие IN_CMD за трансфера и връща модула в състояние „Enabled“.

Алгоритмите, които се изпълняват във всяко състояние, се моделират чрез функционални диаграми. Един от възможните изпълними алгоритми за модула „Storage Control” е представен на фиг.3.12. Диаграмата е разделена на три части, съответно за сензорите,

самото управление и изпълнителните механизми. Действията в частите на сензорите и изпълнителните механизми са специфицирани само чрез стереотипи от тип „send” и тип „receive”. В зависимост от използвания стереотип, действията имат различно графично представяне. За частта, описваща действията на сензорите, действията с имена SUB_Pos_E и SUB_Retract са от тип „receive”, а действията IND_Extend и IND_Retract са от тип „send”.



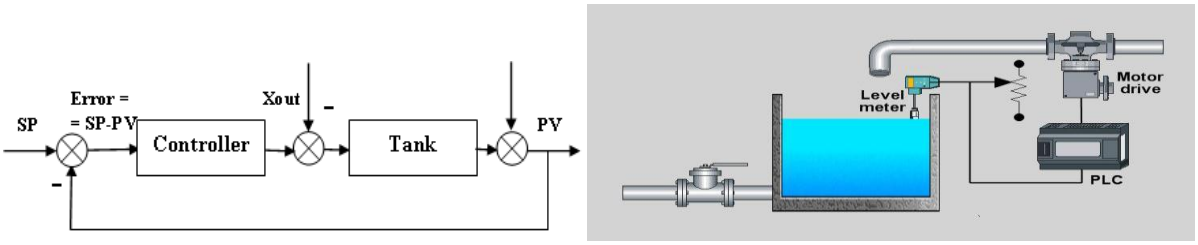
Фиг.3.12: Функционална диаграма за модула„Storage Contrl”

За частта, описваща действията на изпълнителните механизми, действията от тип “send” са “PUB_Extend” и “PUB_Retract”, а тези от тип “receive” са “REQ_Pos_E” и “REQ_Pos_R”. За частта, описваща действията по управлението, освен стереотипи от тип “send” и “receive”, се използват и действия, разпределни към стереотип от тип “condition” и “execution”. Преминаването от едно действие към друго става след удовлетворяване на определено условие. Действията за модула “Storage Control” се извършват в следната последователност: стартиране; инициализация на модула; изчакване; проверка за извършената инициализация; при успешна инициализация се преминава към действие за преместване на пушера, а при неуспешна алгоритъма не се продължава; следващото действие в алгоритъма е преместване на пушера в позиция избутване; проверка на изпълнението; паралелно с това действие се изпълняват действия за визуализиране на резултата от сензора и се изпраща управляващо въздействие към изпълнителния механизъм; действие за проверка преместването на пушера; при изпълнение на условието пушера се връща в изходна позиция, в противен случай проверката се изпълнява отново; едновременно с връщане на пушера в изходна позиция се изпълняват действия за визуализиране на резултата от сензора и се изпраща управляващо въздействие към изпълнителния механизъм; действие за проверка връщането на пушера; при резултат от проверката „лъжа“ се изчаква, а при „истина“, алгоритъмът се повтаря.

Създадените диаграми на състоянията и функционалните диаграми за модула „Transfer Control” са подобни.

3.3. Моделиране на системи за управление на непрекъснати процеси

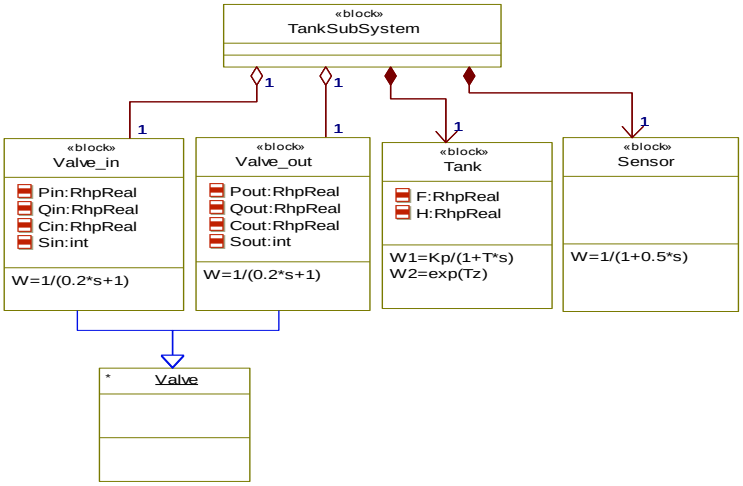
Предложеният в трета глава подход е илюстриран и с пример на моделиране на затворената система за управление на нивото в резервоар, реализираща ПИД закон на регулиране, представен на фиг.3.14-a и 3.14-b, съответно със структурна схема и модел на физическата система.



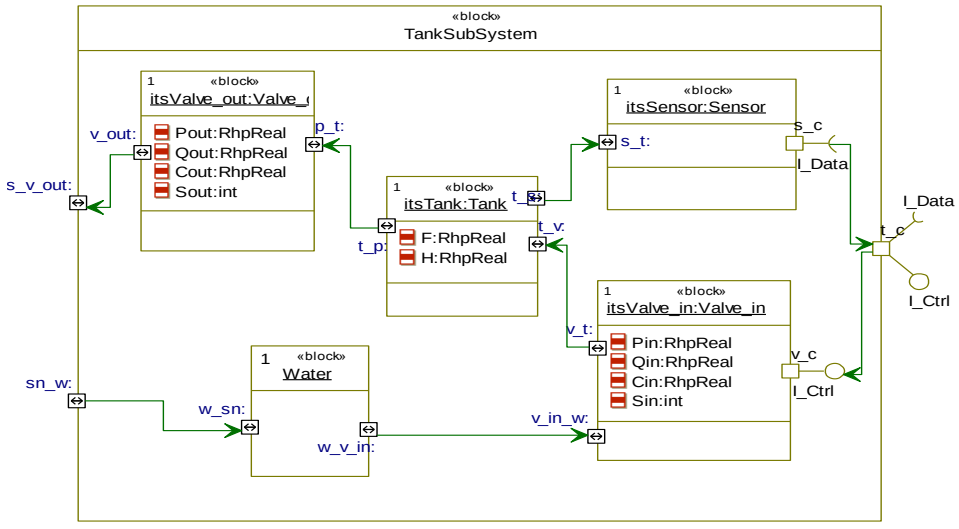
a) b)

Фиг.3.14: Система за управление на ниво

На фиг.3.15 е показана BDD диаграмата на обекта за управление “TankSubSystem”, състояща се от резервоар “Tank”, входен вентил “Valve_in”, изходен вентил “Valve_out” и сензор на ниво “Sensor”. Вътрешната структура на блоковете, включени в BDD, частите и връзките между тях се описва с диаграма на вътрешните блокове (IBD), представена на фиг.3.17. Частите, изграждащи подсистемата на резервоара, са представени с блокове с имена Valve_in, Tank, Sensor и Valve_out.

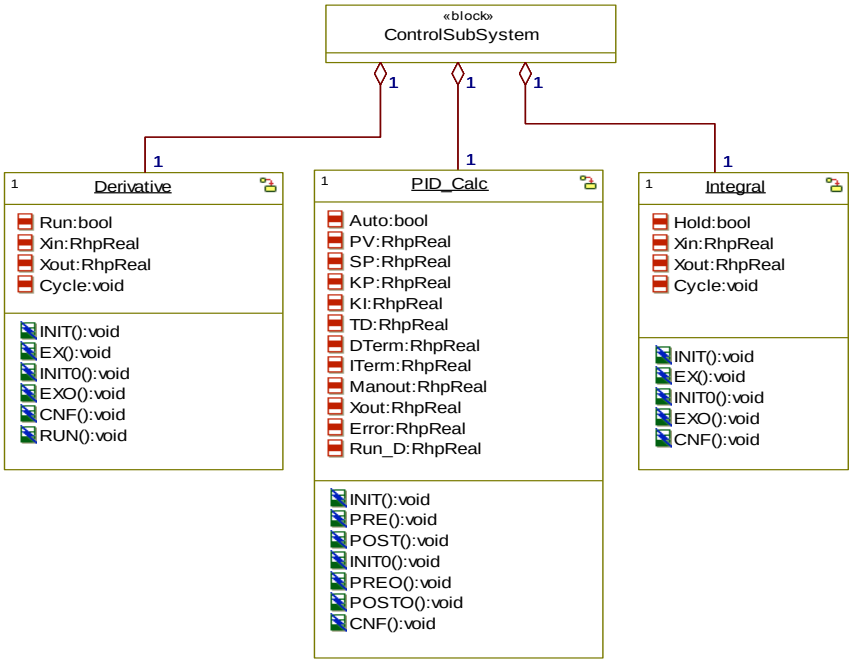


Фиг.3.15: BDD диаграма на статичната съставна структура на физическата подсистема



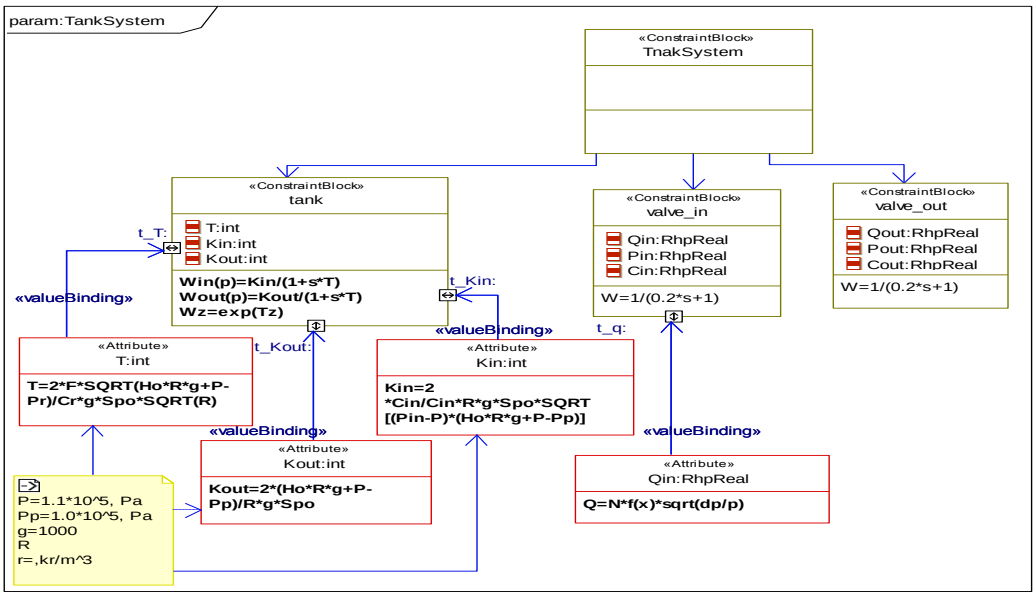
Фиг.3.17: IBD диаграма на системата за управление на ниво

Статичната съставна структура на подсистемата “ControlSubSystem” е реализирана с BDD диаграма, представена на фиг.3.16 и е моделирана чрез блокове с имена Derivative, PID_Calc, Integral и ControlSubSystem. Във всеки един от блоковете, включени в състава на съставния блок, представящ управлението на системата, е вложена диаграма на състоянията, описваща неговото динамично поведение.



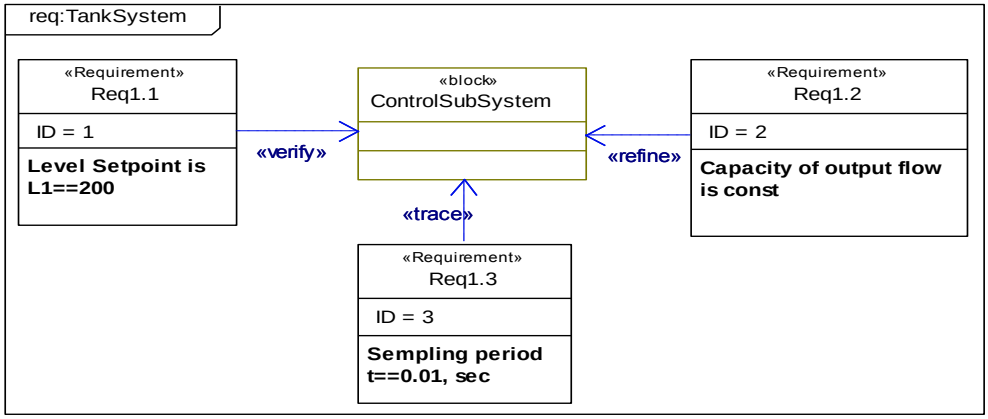
Фиг.3.16: Съставна структура на ПИД контролер в SysML

Параметричните ограничения са добавени в SysML, за да поддържат задачите за анализ и описание на системата с уравнения, характеризиращи физическата система. Параметричните ограничения за подсистемата на резервоара са представени с параметричната диаграма от фиг.3.18 под формата на предавателни функции за резервоара, входния и изходен клапани. Предавателната функция, описваща резервоара включва следните параметри – T, Kin и Kout, зададени като атрибути с дефиниран тип и начин на определяне. Параметрите, включени в предавателните функции, описващи входния и изходния вентили, са Qin, Pin, Cin - за входния вентил и Qout, Pout и Cout - за изходния. Връзката между атрибутите и елементите, изграждащи подсистемата “TanksSubSystem”, се реализира чрез портове от тип „flow”.

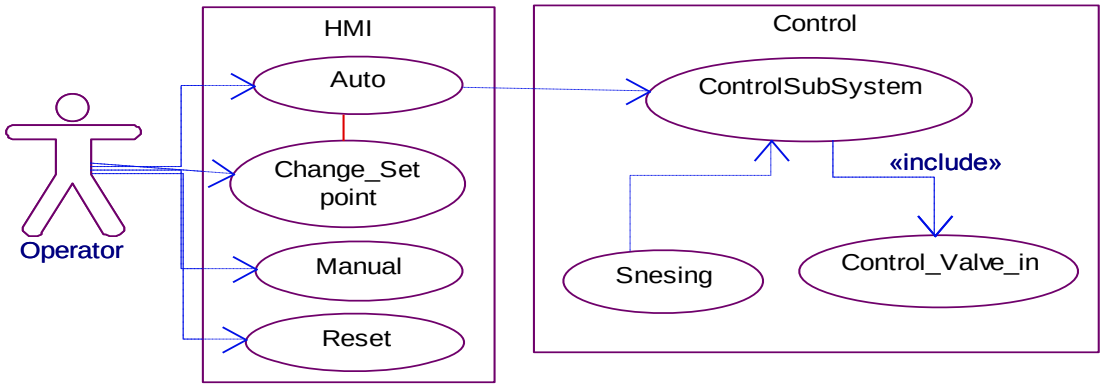


Фиг.3.18: Параметрична диаграма

Функционалните изисквания към подсистемата за управление в диаграмата на изисквания от фиг.3.19 са дефинирани по следния начин: зададеното ниво в резервоара е 200 mm, периодът на обновяване на данните е 0.01 sec, а дебитът на изходния поток е постоянен. Типът на изискването се конкретизира от зададения стереотип на връзката между изискването и частта от системата, към която е предявено. Специфичните функционални аспекти на системата и взаимодействието й с потребителите са представени с диаграмата на случаите на приложение. Един от възможните сценарии е разгледан в диаграмата на случаите на приложение от фиг.3.20. Диаграмата показва, че разглежданата система може да работи в различни режими като автоматичен, ръчен, повторно включване, режим на пълнене, режим на изпразване и измерване, изискващи изпълнение на различна последователност от действия от страна на оператора и контролера.



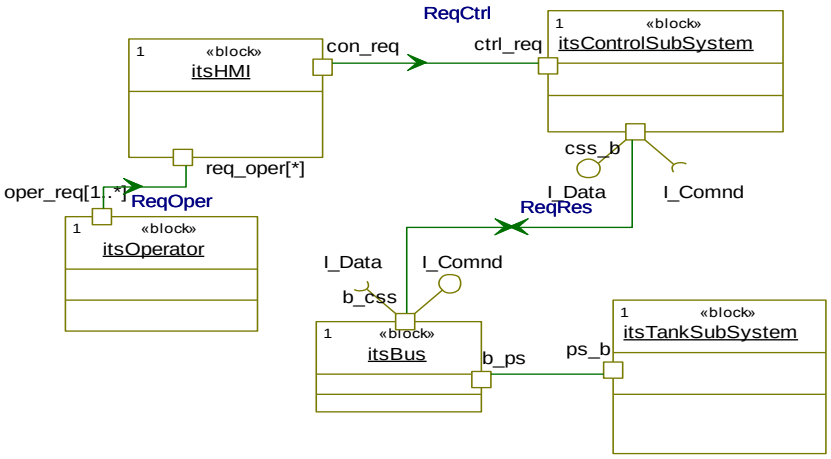
Фиг.3.19: Диаграма на изискванията засистемата за управление на ниво



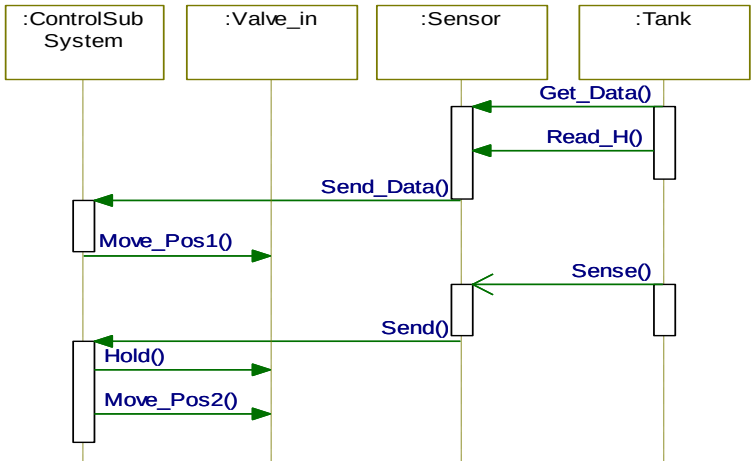
Фиг.3.20: Дефиниране на функционалните изисквания чрез диаграма на случаите на приложение

Статичната структура на системата, състояща се от човеко-машинен интерфейс, подсистема за управление, подсистема на резервоар, комуникационна шина и връзките между тях е представена с BDD диаграма, представена на фиг.3.21. Основните компоненти на системата са моделирани като блокове с имена “ControlSubSystem”, “TankSubSystem”, “HMI”, “Bus” и “Operator”. Комуникацията между тях се осъществява посредством стандартен тип портове, които са специфицирани за дефинирания интерфейс чрез стереотипи “provide” и “require”. Стереотипите от тип “provide” са с имена I_Data, а тези от тип “require” с имена I_Comnd.

Съобщенията, обменяни между системните блокове, се представят с диаграма на последователност от тип „бяла кутия” (WB-SD), представена на фиг.3.22. След измерване на текущото ниво в резервоара, подсистемата за управление прочита тези данни, изчислява управляващото въздействие и отваря или притваря изпълнителния механизъм (“Valve_in”). Съобщенията, представящи управлението на клапана за входния поток “Valve_in” са “Move_Pos1”, “Hold” и “Move_Pos2” и зависят от нивото в резервоара.



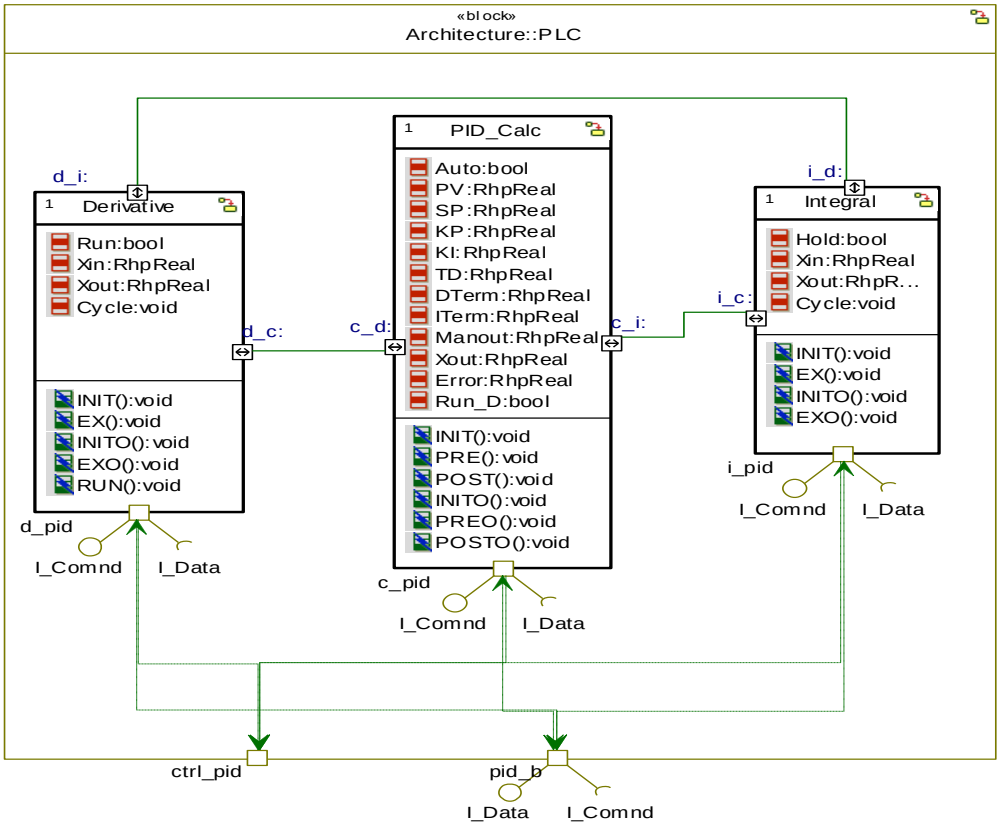
Фиг.3.21: Изглед на статичната структура на системата



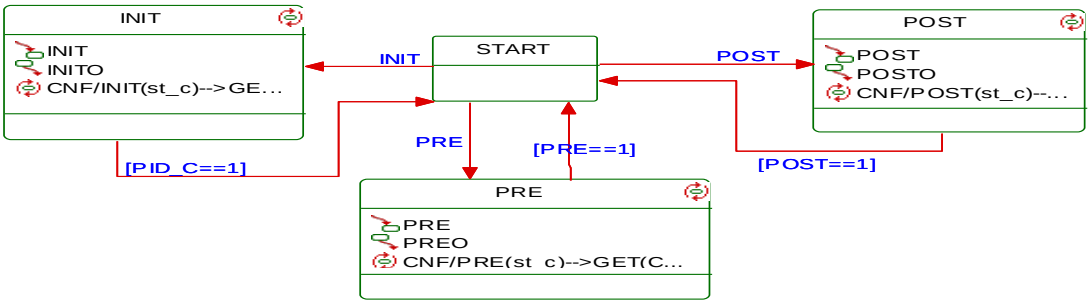
Фиг.3.22: Диаграма на последователност от тип „бяла кутия”

Подсистемата за управление се представя с един системен блок с име “ControlSusbSystem”, а вътрешната му структура се представя с диаграма на вътрешни блокове (фиг.3.23), състояща се от три основни части “PID_Calc”, “Derivative” и “Integral”, представящи съответно пропорционалната, диференциращата и интегриращата съставящи на ПИД регулатора. Връзките между частите, изграждащи подсистемата за управление са представени с портове за потоци. Частта “PID_Calc” капсулова ПИД алгоритъма и изходните стойности, които са пропорционални на отклонението на управляемата променлива от зададената стойност, грешката от интегриращата и диференциращата съставящи. Поведението на частите, реализиращи управлението се представя с атрибути и операции.

Поведението на разглежданата система се моделира с диаграми на състояние и функционални диаграми. При изпълнението на ПИД алгоритъма, поведението на пропорционалната, интегриращата и диференциращата съставящи се описва с диаграми на състоянието. Частта PID_Calc има три състояния (фиг.3.24) – INIT_C и двете изпълними състояния наречени PRE и POST. Първото състояние изпълнява алгоритъм за изчисляване на грешката между заданието и измерената стойност. Генерирането на изходно събитие PREO води до целево изпълнение на алгоритмите в диференциращата и интегриращата части. Второто състояние POST се достига при активизиране на входно събитие със същото име, получено след изпълнението на алгоритмите в интегриращата и диференциращата части. Всяко състояние се характеризира с входно действие, изходно действие и един или няколко прости алгоритъма, изпълнявани в него. За да се премине от състояние „Start”, в състояние „Init”, „PRE” или „POST”, трябва да са изпълнени условията в преходите. Създадени са диаграми на състояния и за интегриращата и диференциращата съставящи.

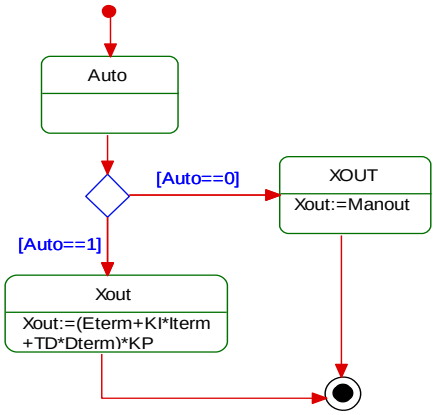


Фиг.3.23: IBD на модела на ПИД контролера



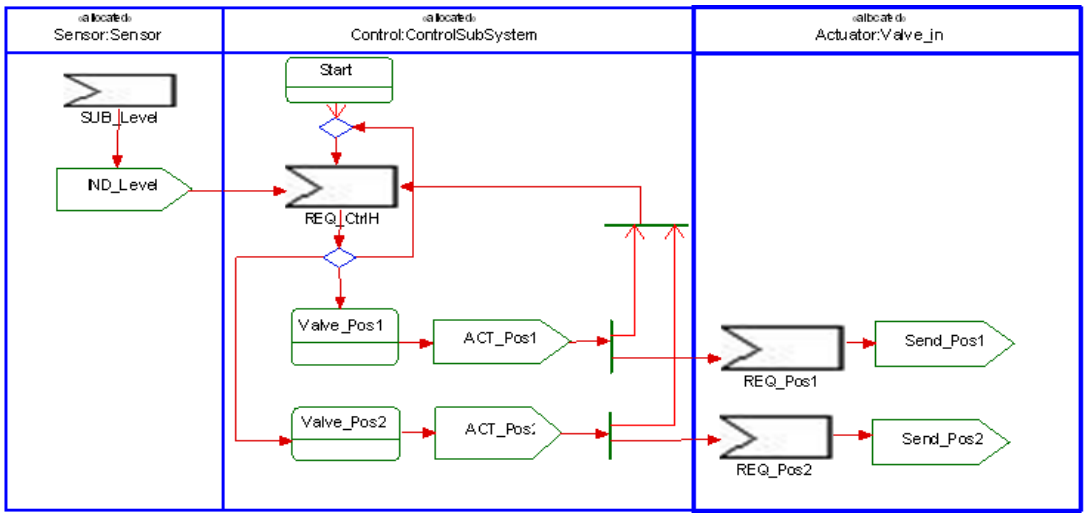
Фиг.3.24: Диаграма на състояния за пропорционалната съставяща

Алгоритмите, изпълнявани в отделните състояния се моделират с използване на функционални диаграми. Примерен алгоритъм за пропорционалната съставяща е представен на фиг.3.27. Алгоритмите, изпълнявани в трите съставящи на ПИД контролера, са за обновяване на стойността на изхода X_{OUT}.



Фиг.3.27:Функционална диаграма представяща алгоритъм, изпълняван в пропорционалната съставяща на ПИД контролера

Следващият етап от методологията е разпределяне на софтуерните елементи към съответните хардуерни устройства. Това разпределяне в SysML се осъществява, използвайки алокация. Алокацията осигурява обща връзка за разпределяне на елемент от модела към друг, включвайки специфични подкласове, които ограничават използването им – структурни, поведенчески и потоци. Алокацията на поведение е използвана да разпредели поведението към блок, реализиращ това поведение. За целта е използвана функционална диаграма, представена на фиг.3.30, където всяко действие е разпределено към част от статичната структура на системата. Диаграмата е разделена на три части чрез плаваща линия, като всяка от частите съответства на една от подсистемите в системата за управление на ниво.



Фиг.3.30: Алокация на поведение

3.4. Изводи и заключения към глава трета

1. В тази глава е предложен подход за разработка на системи за управление с използване на UML профила за системно инженерство SysML и програмния продукт Rhapsody. Разработеният подход се отличава със следните предимства:
   - Чрез SysML се поддържа целия жизнен цикъл на системата за управление, започвайки от дефинирането на изискванията до софтуерната имплементация;
   - Създадена е възможност за детайлно моделиране на обекта за управление (физическата система), като се подпомага прилагането на методи за анализ, тестване и валидация на проектираната система.
2. Предложеният подход е приложен за моделиране, както на разпределени, така и на непрекъснати производствени системи.
   - Продуктът Rhapsody позволява да се генерира код на създадения модел на един от следните програмни езици: C, C++, Java и Ada. Въведени са и средства, позволяващи ранното откриване на грешки при проектирането на приложения, а също така и възможности за специфициране на приложенията за конкретни потребителски решения или тяхната интеграция с други платформи.
3. Недостатъците на предложения подход и използвания програмен продукт са:
   - Подходът е приложим за моделиране на разпределени системи за управление, въпреки че не осигурява всички свойства на тези системи.
   - Rhapsody не позволява влагане на функционални диаграми, представящи изпълнимите алгоритми в съответните състояния от диаграмата на състояния.
   - На този етап на разработка, програмният продукт не позволява и тестване чрез симулация на модела.

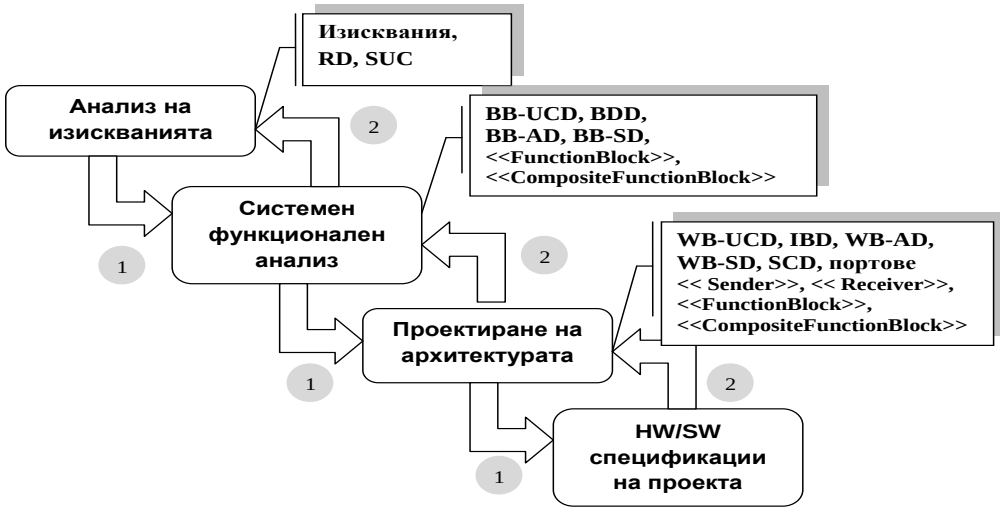
4. Подходът може да бъде приложен и за разработка на хибридни системи чрез използване на разширени функционални диаграми, параметрични диаграми и портове на потоци с техните части.

С оглед на съвременните тенденции в развитието на системите за управление и по-конкретно за целите на моделно управляваната разработка на разпределени системи за управление, в следващата глава на дисертацията е предложено разширение на предложения в тази глава подход, на базата на комбинацията между UML/SysML и референтната архитектура и модели на стандарта IEC-61499.

IV. Комбиниран подход за разработка на разпределени системи за управление на базата на UML/SysML и стандарта IEC-61499

4.1. Представяне на комбинирания подход

Предлаганият подход, обединяващ предимствата на UML/SysML профила и стандарта IEC-61499 представялва продължение на подхода, базиран само на UML/SysML профила, разгледан в предишната глава, като тук са добавени нови нотации към различните етапи на разработка, съобразени с моделите, дефинирани от стандарта IEC-61499. Той дава възможност за бърза и навременна реконфигурация на системата в условията на изменения в продуктовата гама и настъпили вътрешни смущения, което води до по-детайлно и пълно отразяване на изискванията към системата за управление, на нейната структура, поведение и физическа реализация. Комбинираният подход, илюстриран на фиг.4.1 добавя нови модели към различните етапи на разработкана предложения в трета глава подход, съобразени с моделите, дефинирани от стандарта IEC-61499.



Фиг.4.1: Илюстрация на комбинирания подход

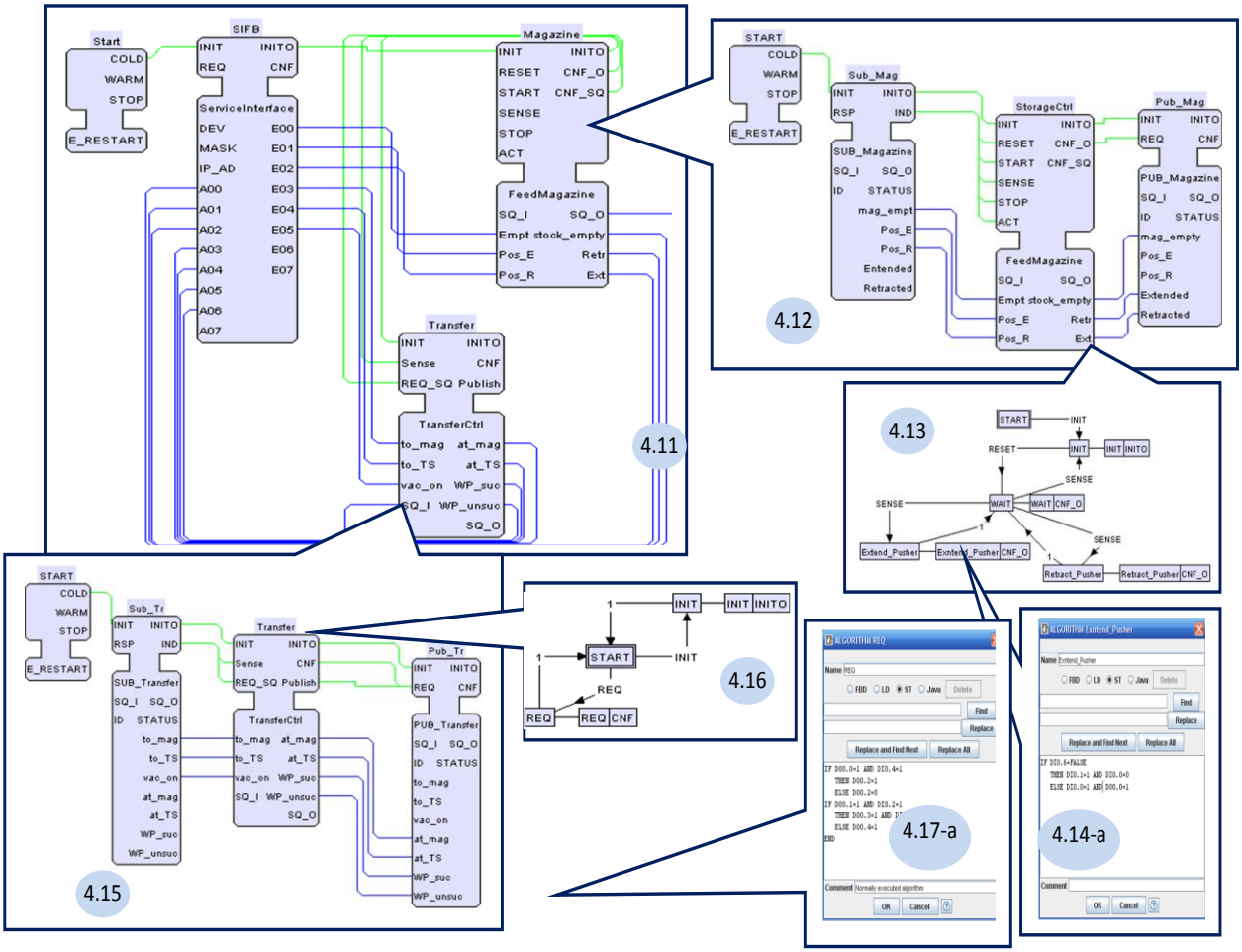
За етапа на системен функционален анализ, с цел моделирането на многократно използваеми, отворени и независими от потребителя приложения за управление, са дефинирани два допълнителни стереотипа – „FunctionBlock” и „CompositeFunctionBlock”, съответстващи на референтните модели, съгласно стандарта IEC-61499. Чрез присвояването на тези стереотипи към системните блокове от диаграмата за дефиниране на блокове (BDD), структурата на системата се представя на различни нива на интеграция - устройство, ресурси и приложения. Функционалната диаграма (BB-AD) и диаграмите на последователност (BB-SD), които са диаграми от тип “черна кутия” се използват за моделиране на функционалността на системата на високо ниво на абстрактност.

Проектирането на архитектурата е третия етап на разработка от методологията и включва проектиране структурата на подсистемите и поведението, базирано на диаграми, използвани на предходния етап, но използващи варианта на „бялата кутия” - (WB-UCD,

WB-AD, WB-SD). Дефинираните системи се представят от части (parts), съответстващи на основния градивен елемент на системата за управление – функционалния блок. Частите са вложени в системните блокове, като им се присвояват стереотипи „FunctionBlock", а на системните блокове – „CompositeFunctionBlock". За да идентифицира интерфейса на физическата система, се дефинират портове, използващи интерфейс, дефиниран чрез стереотипите „Sender" и „Receiver". Интерфейсът „Receiver" характеризира заявките от външни условия, а интерфейса „Sender" характеризира заявките от порта.

4.2. Приложение на стандарта IEC-61499 при разработката на системи за управление

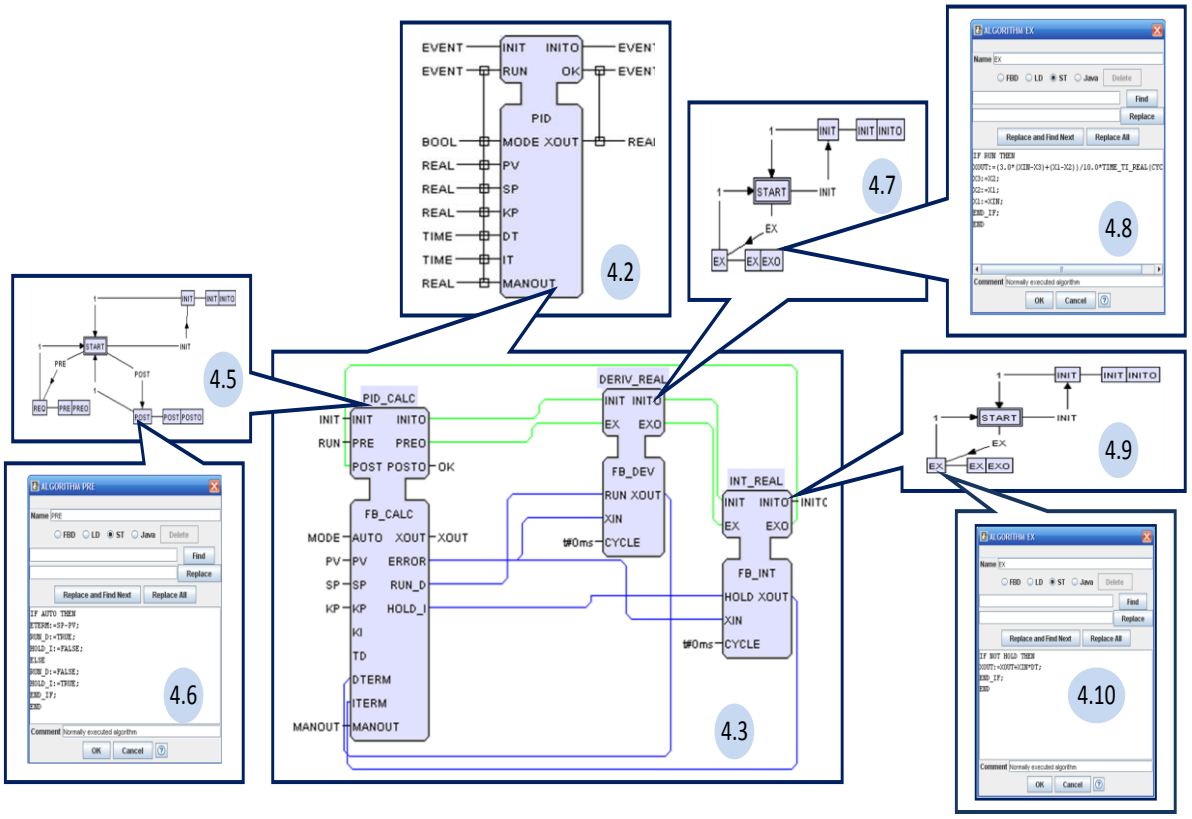
Приложението на стандарта IEC-61499 при разработката на разпределени системи за управление е обобщено на фиг.О.6. Станцията за разпределяне на детайли е моделирана посредством диаграма на функционални блокове (ФБ) (фиг.4.11). Двата модула на станцията – „Storage Control" и „Transfer Control" са реализирани чрез два съставни функционални блока, чиято структура е представена чрез диаграми на ФБ на фиг.О.6-4.12 и фиг.О.6-4.15. Капсулираното вътрешно поведение на тези ФБ е моделирано чрез графи за управление на изпълнението (ECC), представени на фиг.О.6-4.13 и фиг.О.6-4.16, а асоциираните към графите за управление на изпълнението алгоритми са представени на фиг.О.6-4.14-а и фиг.О.6-4.17-а.



Фиг.О.6: Разработка на разпределена система за управление на базата на IEC-61499

Приложението на стандарта IEC-61499 при разработката на система за управление на непрекъснати процеси е представено на обобщената фиг.О.7. Контролерът, реализиращ ПИД закон на регулиране и базиран на стандарта IEC-61499 е представен на фиг.О.7-4.2 като съставен функционален блок, включващ мрежа от три екземпляра на основните ФБ

FB_Calc, FB_INT и FB_DEV, които са от тип PID_Calc, INTEGRAL_REAL и DERIVATIVE_REAL, както е показано на фиг.О.7-4.3. Графът за управление на изпълнението на блока PID_Calc е представен на фиг.О.7-4.5, за блока FB_DEV на фиг.О.7-4.7 и за FB_DEV на фиг.О.7-4.9. Примерни алгоритми за изпълнение в състоянията са показани на фиг.О.7-4.6, О.7-4.8 и фиг.О.7-4.10, съответно.



Фиг.О-7:Разработка на система за управление на непрекъснати процеси на базата на стандарта IEC-61499

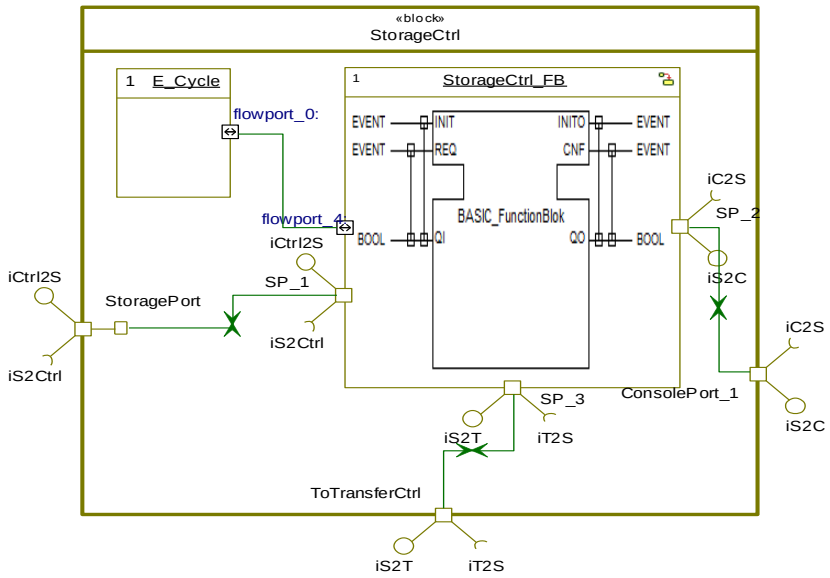
4.3. Приложение на комбинирания подход при разработката на системи за управление

Подходът е илюстриран с пример на разпределена система за управление на “Станция за разпределяне на детайли”. Първият етап при проектирането на системата за управление е описан подробно в трета глава и се състои в дефиниране на функционалните изисквания към системата, което е реализирано с диаграмата на изисквания и диаграмата на случаите на приложение, отразяваща взаимодействието между системата и нейните потребители, както и изискванията към разпределената система за управление. Моделирането на статичната съставна структура на системата, състояща се от контролер, конзола, шина и самата станция за разпределяне ена базата на BDD диаграма (фиг.3.5). Вътрешната структура на станцията е моделирана, както с BDD диаграма, представяща йерархията в системата, така и с IBD диаграми. Вътрешната структура на системните блокове е моделирана, използвайки BDD и IBD диаграми. Съобщенията, обменяни между системните блокове са представени чрез диаграмата на последователност и функционална диаграма, за да обхванат различни сценарии. Поведението на системата е моделирано с функционални диаграми и диаграми на състоянието.

При втория етап от проектирането на системата, двата разпределени контролера за управление на станцията за разпределяне на детайли се моделира, съгласно стандарта IEC-61499. За тази цел в дървовидната структура на модела, в работната среда на Rhapsody е създаден допълнителен профил „SysML-IEC61499“. По този начин към съществуващите свойства и ограничения на SysML профила се добавят и изисквания, свойства и

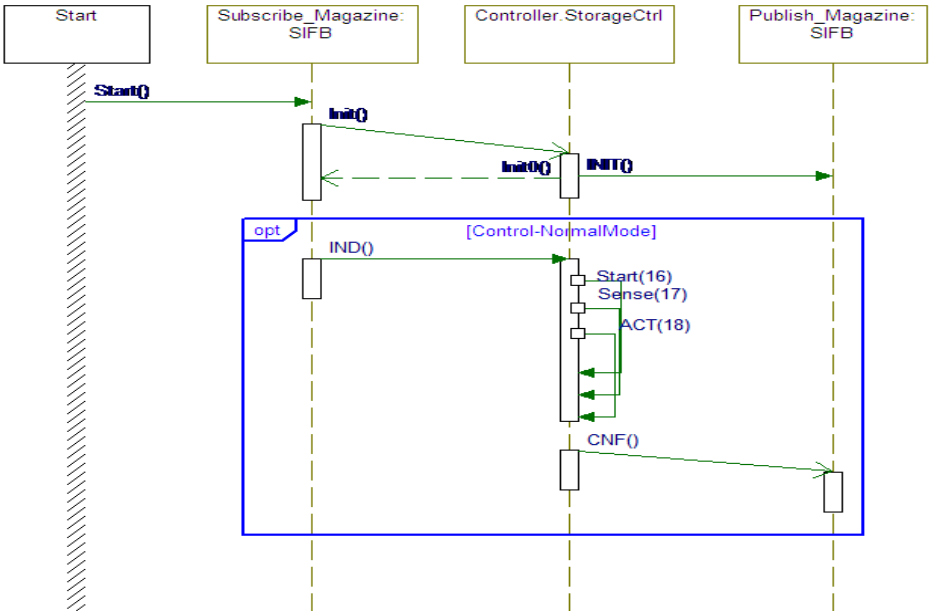
ограничения, базирани на стандарта IEC-61499. Дефинирани са допълнителни стереотипи като „BasicFunctionBlock”, „CompositeFunctionBlock” и др., съгласно основните понятия и модели на стандарта.

По-сложни структури на ресурса могат да се представят чрез IBD диаграми, които при определени условия са еквивалентни на съставни функционални блокове или на подприложения. На фиг.4.17 е представена структурата на контролера за управление на магазина (StorageCtrl), моделиран с IBD диаграма в UML/SysML. Контролерът е представен като системен блок със стереотип „BasicFunctionBlock”.



Фиг.4.17: Диаграма на вътрешни блокове на контолера „StorageControl”

Диаграмата на последователност изпълнява съществена роля при определяне на входно/ изходните данни и събития в диаграмата на ФБ (FBD). Входните съобщения за даден системен блок съответстват на входните събития за ФБ. Съответно изходните съобщения на този клас са изходните събития за ФБ. Параметрите характеризиращи входно/изходните съобщения са входно/изходните данни на ФБ. Диаграмата на последователност за модула „Storage Control”е представена на фиг.4.19 и се състои от четири обекта с имена Start, Subscribe_Magazine:SIFB, Controller:StorageCtrl и Publish_Magazine:SIFB. По аналогичен начин е моделирана и диаграмата на последователност за модула „TransferControl”.



Фиг.4.19: Диаграма на последователност за модула “StorageCtrl”

Графът за управление на изпълнението за модула „Storage Control” е представен в глава трета на фиг.3.10, а на фиг.3.11 е представена функционалната диаграма, реализираща един от възможните алгоритми, асоцииран към графа за управление на изпълнението за модула.

4.4. Изводи и заключения към четвърта глава

1. В тази глава е предложен комбиниран подход за моделиране на разпределени системи за управление, базиран на UML/SysML и стандарта IEC-61499. Изводите, които могат да бъдат направени са:
   - Разработените приложения, чрез използване на двата стандарта поотделно и в комбинация, поддържат целия жизнен цикъл за моделиране на една софтуерна система – от анализа и дефиниране на изискванията към системата до имплементацията.
   - Целият инженерингов процес при проектирането на разпределени системи е подобрен по отношение на надеждност, многократна използваемост и оперативна съвместимост.
   - Значително се съкращава времето за моделиране на различни системи, поради възможността за многократно използване на вече дефинирани ФБ.
2. Предложеният подход е тестван за моделиране на примерни системи за управление: на непрекъснати и дискретни процеси.

V. Разработка на агентно-базирани системи за управление на базата на UML/SysML

Концепцията на мулти-агентните системи се оказва перспективно направление за компенсиране недостатъците на обектно-ориентирания подход чрез прилагане на принципите на агентно-ориентираното софтуерно инженерство (AOSE). Едно от направленията на изследователската общност в областта на AOSE са различни разширения на UML за подпомагане процесите на моделиране на основните агентно-базирани концепции, като агенти, онтология и протоколи за взаимодействие.

С развитието на UML и въвеждането на активни обекти, разликите между обектите и агентите намалява, благодарение на подобряване на мета-модела на езика, формализация на различни диаграми, използвани от езика и въвеждането на нови нотации. Това позволява постигането на по-висока степен на отразяване на основните свойства на агентите и техните взаимодействия. В сравнение с други алтернативи, основното предимство от прилагането на UML при разработката на MAS е, че използването на стандартни механизми и спецификации, е от решаващо значение за осигуряване на оперативна съвместимост между автономни единици в една отворена и разпределена агентно-базирана интегрирана среда за управление.

5.1. Анализ на разширенията на UML за моделиране на агентно-ориентирани системи за управление

За разработка на агентно-ориентирани системи се използват различни методологии като: езика AUML, езика ACL, допълнителни разширения и стереотипи в структурата на UML, стандарта FIPA, протокола за комуникации между агенти AIP, методологиите MAS-COMMONKADS, MAS-SIVE, MaSE, MESSAGE, INGENIAS, GAIA и TROPOS. Важна роля в тази насока играят консорциумите OMG и FIPA, които имат за цел да стимулират възприемането на агентно-базираните технологии в индустрията чрез обединяване на различни стандарти за разработка на обектно-ориентиран софтуер с инициативите на FIPA.

Стандартите UML и FIPA предлагат възможност за пълна спецификация на жизнения цикъл на една агентно-базирана система. Тази възможност се реализира чрез разширяване на UML с протоколи за комуникации между агенти AIP.

5.2. Подход за моделиране на агентно-базирани системи за управление на база стандартите UML/SysML и IEC-61499

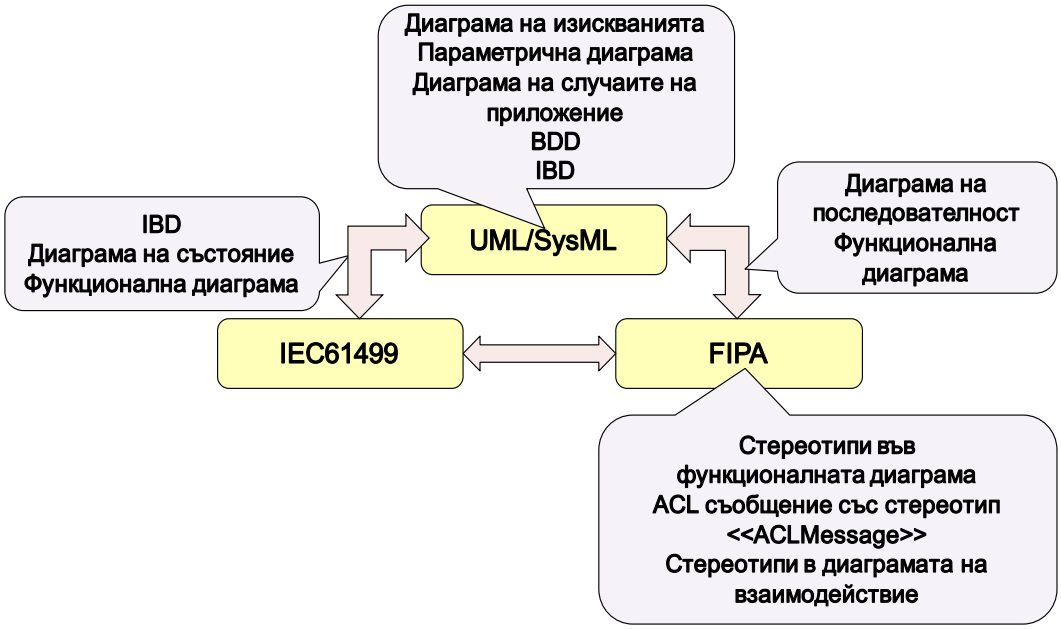
За разработката на мулти-агентни системи с използване на UML е предложен нов подход, базиращ се на комбинирания подход за разработка на разпределени системи за управление, представен в Четвърта глава на дисертационния труд и разширен с референтния модел на FIPA. Основната идея на подхода е свързана със създаването на моделно управляваната разработка на агентно базирани системи за управление, отразяваща адекватно дефинираните към системата изисквания и осигуряваща възможност за формална спецификация, верификация и валидация на използваните по време на жизнения цикъл модели. В тази връзка към представените в Четвърта глава допълнителни стереотипи към профила UML/SysML, отразяващи референтната архитектура и модели на стандарта IEC-61499 са добавени нови такива, свързани с постигане на агентно базираните свойства на обектите на базата на протоколите на FIPA за комуникация между агентите. Чрез използване на протоколите на FIPA в диаграмите се разширява семантиката на съобщенията, постига се паралелност на съобщенията, вложеност на протоколите и образците на протоколи, като по този начин свойствата на обекта се доближават до тези на агента. По този начин се предлага комбинираният подход, съвместяващ комбинираното използване на профила UML/SysML и стандарта IEC-61499 да бъде използван за моделиране на вътрешната структура на системите за управление и дефиниране на това как тази система може да бъде разпределена към физическите единици, докато при моделиране на комуникациите между човеко-машинния интерфейс (HMI) и системите за управление, и между самите тях се предлага вграждане в използвания профил на подходящо подбран референтен модел на FIPA.

На фиг.5.1 са илюстрирани основните етапи на предложения подход, базиращ се на комбинация от посочените по-горе подходи. Първият етап при разработката на системи за управление, използвайки предложения подход, е създаване на UML/SysML модел, базиран на стандарта IEC-61499. За тази цел е използван предложения в четвърта глава, раздел 4.1 софтуерен процес. Физическата система е описана чрез параметричната диаграма от SysML профила. Използвайки диаграми на случаите на приложение, диаграми за дефиниране на блокове и диаграми на вътрешни блокове, които са елементи от същия профил, могат да бъдат моделирани различни гледни точки на системата за управление. Системната архитектура се проектира чрез използването на BDD и IBD диаграми. Етапът на функционален анализ се реализира с разработката на диаграми на последователност, функционални диаграми и диаграми на състояние. Изискванията към системата са зададени с диаграма на изискванията чрез създаване таксономия на изискванията, а също и чрез диаграмата на случаите на приложение.

Вторият етап на подхода е създаване на профил, базиран на стандарта IEC-61499. В него се създават моделите на човеко-машинния интерфейс и модела на системата за управление. Диаграмите, използвани за създаването на тези модели са IBD, диаграми на състоянията и функционални диаграми, като за всеки елемент от диаграмите е дефиниран допълнителен стереотип, съгласно стандарта IEC-61499.

Разширяването на модела със стандартни комуникационни модели е третия етап на предлагания подход. Избран е референтният модел на комуникационния протокол FIPA, защото той осигурява нормативна рамка, където агентите могат да се проявяват и действат. Спецификацията на FIPA въвежда логически референтен модел за създаване, регистрация, определяне на местоположението, комуникация, миграция и разрушаване на агенти. Стандартът FIPA не се опитва да определи вътрешната архитектура на агентите,

нито определя как те трябва да бъдат изпълнени, а специфицира необходимия интерфейс за поддържане на оперативна съвместимост между агентите в системата.



Фиг.5.1: Илюстрация на подхода за разработка на агентно-базирани системи за управление

5.3. Приложение на подхода при разработката на разпределена система за управление

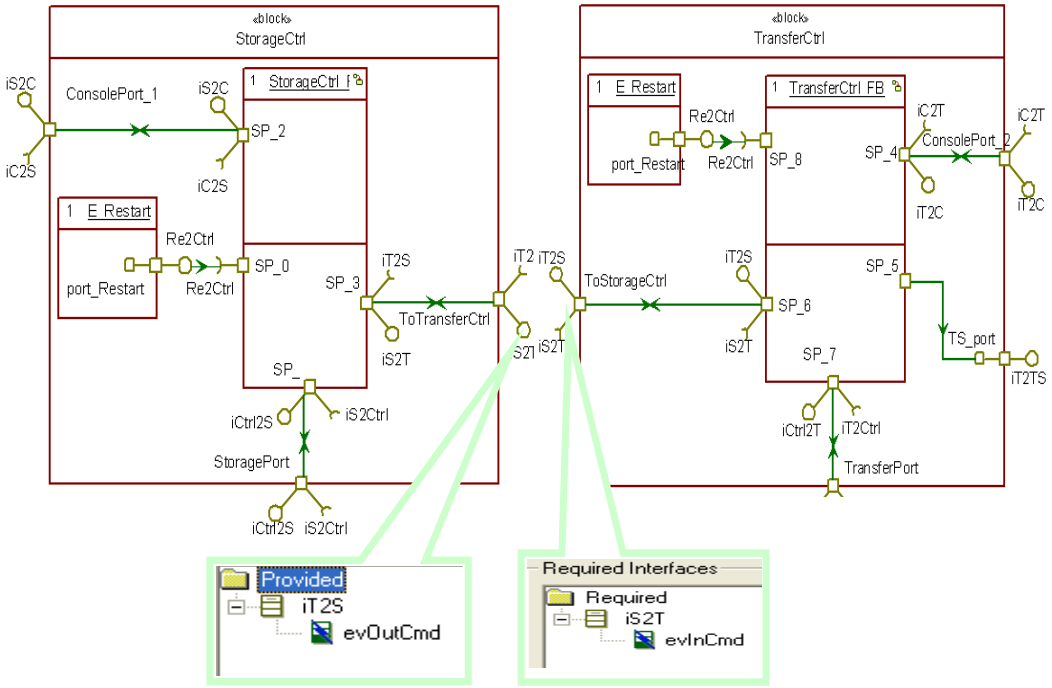
Предложеният подход е илюстриран с примера на разработката на разпределена система за управление на FESTO станцията за разпределяне на детайли. Тъй като подходът е разширение на подходите, описани в глава три, раздел 3.1. и глава четири, раздел 4.2, които са онагледени със същия пример, то в по-голямата си част моделно управляваната разработка и използваните модели не се различават от моделите, разработени съгласно предишните подходи.

Предложеният подход оказва влияние върху моделирането на комуникациите, като комуникацията между двата модула, съгласно стандарта IEC-61499, се реализира посредством функционални блокове от тип “Service Interface Function Block”. Комуникационният интерфейс в UML2.x се реализира чрез различен тип портове, използвани, за да дефинират свързването между входните и изходните компоненти на системата. На фиг.5.2 са представени комуникационните портове между контролерите, използващи интерфейси от тип “provided” и “required”. Интерфейсът “provided” характеризира заявките отвън-навътре, а интерфейсът “required” характеризира заявките отвътре - навън. Портовете от тип “required” се свързват с портове от тип “provided”.

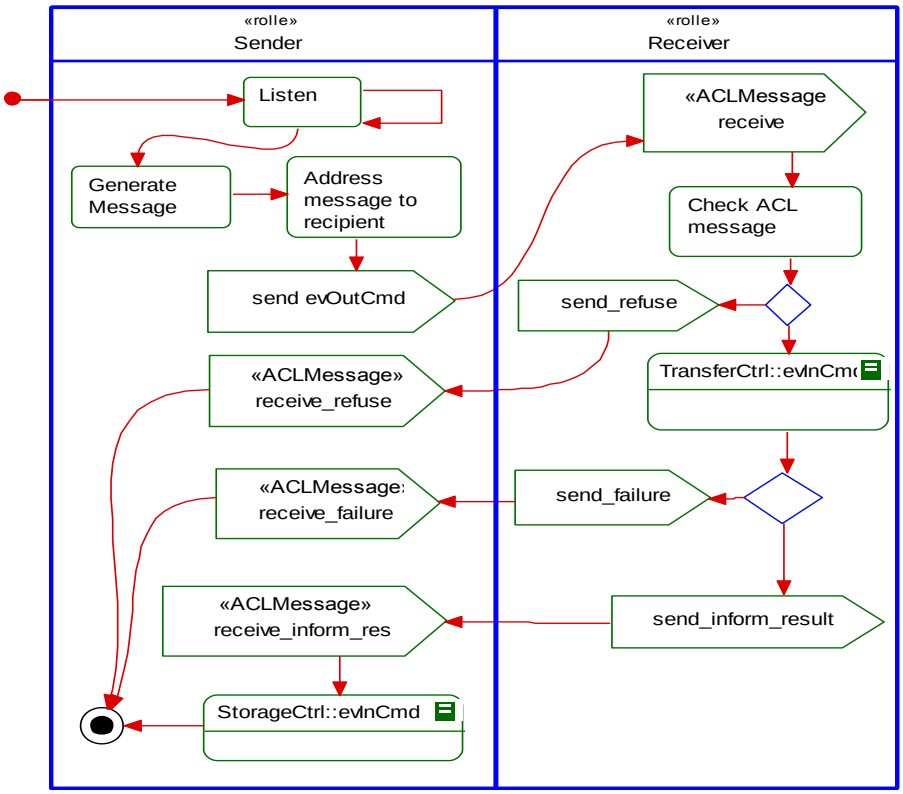
Протоколът FIPA е използван като „заявка” за обмен на информация от типа всеки към всеки между контролерите, с цел да получи информация за изпълнените задачи между агентите. Взаимодействията в протокола са моделирани чрез функционални диаграми (фиг.5.3), като поведението на използвания интерфейс е показано чрез стереотипа <<role>>.

Във функционалната диаграма действията от тип “sender” и “receiver” са базирани на комуникационни протоколи и позволяват на агентите да разменят съобщения. Основното предимство от използването на протокола FIPA за комуникации между агенти е, че той съдържа унифицирани съответствия за методите, използвани за инициализация и приключване на съобщенията, форматиране и кодиране на съобщения, синхронизация между получател и подател и откриване и коригиране на грешки от пренасянето на

информация. Спецификацията на протокола се състои от следните части: спецификация за услугата, характеристики на околна среда, точно определен формат на съобщението (синтаксис), процедурни правила за обмен на данни (граматика) и речник на съобщенията заедно с тяхното значение (семантика).



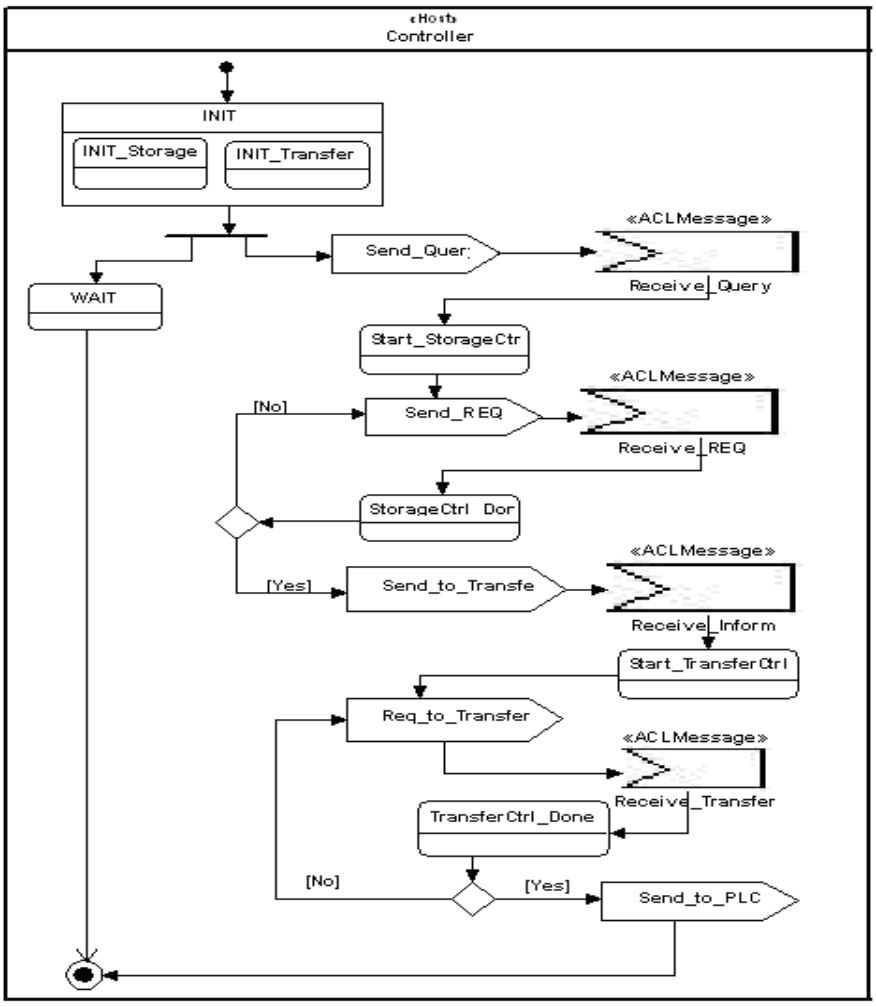
Фиг.5.2: Комуникация между разпределените контролери



Фиг.5.3: Функционална диаграма за поведение на интерфейс

Комуникацията между управляващия софтуер от една страна и сензорите, и изпълнителните механизми от друга, които са обектни компоненти, съгласно IEC-61499, е реализирана чрез SIFB. В UML2.0 комуникационният интерфейс се дефинира с различни типове портове. Станцията за разпределяне на детайли използва комуникационна шина

тип PROFIBUS и според SysML профила, комуникационният интерфейс се представя със стандартен тип портове и портове за потоци. На фиг.5.4 е представена функционална диаграма, представяща комуникацията между управляващия софтуер и човеко-машинния интерфейс, реализирана чрез UML разширенията за FIPA протокола. На базата на референтния модел на стандарта FIPA и езика за комуникации между агенти ACL, в модела е дефиниран нов стереотип <<ACLMessage>>, представящ обмена на съобщения между агентите. Във функционалната диаграма на действията от тип "sender" и "receiver" е присвоен стереотип <<ACLMessage>>, защото са базирани на комуникационни протоколи.



Фиг.5.4: Функционална диаграма реализирана чрез FIPA протокол

5.4. Изводи и заключения към пета глава

1. Създаден е и приложен софтуерен процес за разработката на агентно-базирани системи за управление, гарантиращ моделиране на специфичните характеристики на агентите като автономно поведение и кооперативност със средствата на обектно-ориентираните подходи и UML.
   - Подходът обединява предимствата на три методологии: референтен модел за моделиране на комуникациите, базиран на FIPA, UML профила за системно инженерство – SysML и стандарта за разработка на разпределени системи за управление IEC-61499, дефиниращ основните понятия, референтна архитектура и модели за тяхната разработка.
   - Съществена характеристика на предложения подход е, че инженерите по управление са в състояние да моделират системата за управление и да приложат различни техники на анализ, с цел да определят дали моделите

отговарят на изискванията за изпълнение и прогнозиране, без да са необходими задълбочени познания в тези техники.
- Използвайки този подход, разработчиците не се ангажират с изучаване на нов тип диаграми, а използват съществуващите в UML/SysML профила диаграми, което значително улеснява работата им.
- С използването на разширения във функционалната диаграма, базирани на комуникационния протокол FIPA, подходът се доближава до агентно-базираните системи.

2. Разработеният подход е приложен за целите на разработката на разпределена система за управление на FESTO станцията за разпределяне на детайли.

VI. Общи изводи

Анализът на изследваните обектно-ориентирани подходи и UML при разработката на системи за управление показва, че те се приложими за моделиране на системи в различни области на промишлеността. Разширимостта на обектно-ориентираните средства и UML позволява да се използват предимствата на един подход, като към него се добавят свойства и характеристики на други езици, рамки и подходи с цел постигането на конкретни потребителски решения.

И трите предложени обектно-ориентирани подходи за разработка на системи за управление се базират на надеждна концепция за декомпозиция и модулност, отворени са за разширения в случаите на нови изисквания и предизвикателства към системите за управление и автоматизация, подходите са независими от конкретните среди. Важно предимство на разработените подходи е възможността за моделиране и имплементация на капсулирани, многократно използваеми компоненти с различна степен на автономност и кооперативност и представляват важна стъпка към постигане на реконфигурируемост на системите за управление.

Предложените в дисертационния труд три нови обектно-ориентирани подходи, базирани на използването на UML, са съвместими, интегрално свързани помежду си и могат да бъдат разглеждани като разширения, получени или чрез дефиниране на нови нотации, или посредством използване на допълнителни профили с цел отразяване на различни механизми за поддържане на реално-времеви характеристики, проектиране на отворени, компонентно-базирани разпределени системи, поддържане целия жизнен цикъл на разработка на софтуерни системи, моделиране на йерархични структури на системите, възможност за декомпозиция, детайлно моделиране на физическата система (обекта за управление), пълно отразяване на изискванията както към системата за управление, така и към обекта на управление.

Използването на конкретен подход зависи от изискванията, които се предявяват към конкретната система за управление по отношение на използваната архитектура (централизирана, разпределена), степен на автономност или на надеждност, от вида на обекта за управление (дискретен, непрекъснат, периодичен).

Подобряването на предложените подходи е възможно в няколко посоки: по отношение на повишаване на ефективността на създадения „UML/SysML-IEC61499” профил; по отношение на повишаване на надеждността и сигурността на разработката, посредством интеграцията на ефективни средства за ранна верификация и валидация на моделите и създаваните приложения; по отношение на средствата и подходите за автоматична трансформация на използваните модели и тяхната имплементация в крайни софтуерни продукти.

ПРИНОСИ НА ДИСЕРТАЦИОННАТА РАБОТА

1. Направена е класификация на подходите за моделиране на системи за управление, базирани на UML. Подходите са обединени в два основни клъстера, според възможностите и средствата, които се използват за създаване на различни механизми за поддържане на реално-времеви характеристики като: модели на физическо време, времева спецификация, времеви устройства, моделиране и управление на физически ресурси и паралелност.
2. Разработен е подход за детайлно моделиране на физическа система, който се поддържа само частично от чистия UML. За целта е използван профила на UML за системно инженерство – SysML.
3. Предложен е софтуерен процес за разработка на разпределени системи за управление на базата на модификация на методологията в областта на системното инженерство - Harmony SE. Софтуерният процес поддържа целия жизнен цикъл на разработваната система за управление, започвайки от дефинирането на изискванията до софтуерната имплементация, като се моделира подробно и физическата система.
4. Създаден е изпълним функционален модел на софтуер за ранните етапи на разработка на модел на софтуерна система за управление, базиран на спецификационните изисквания, възможността за формална спецификация, верификация и валидация на системата, обединяващ предимствата на стандартите UML и IEC-61499.
5. Като перспективно направление за компенсиране недостатъците на обектно-ориентирания подход е предложено използване на концепцията на мулти-агентните системи чрез прилагане принципите на агентно-ориентираното софтуерно инженерство (AOSE).
6. Предложен е подход за създаване на изпълним функционален модел на софтуер за ранните етапи на разработка на процес, базиран на спецификационните изисквания, възможността за формална спецификация, верификация и валидация на системата. Подходът обединява предимствата на три методологии, използвайки референтен модел базиран на FIPA, първият формален профил в областта на системното инженерство – SysML и стандартът IEC-61499, дефиниращ основните понятия и методологията за проектиране на модулни, многократно използваеми, отворени и независими от потребителя, разпределени приложения за управление. Предложеният подход е реализиран с тестов пример.

СПИСЪК С ПУБЛИКАЦИИ ПО ДИСЕРТАЦИЯТА

1. Batchkova I., **Antonova I**. (2004), Information System Modeling using UML and IBM Rational Rose, Proceedings of International Conference Automatics and Informatics, pp. 97÷100, October 2004, Sofia, Bulgaria.
2. **Антонова И**., Бачкова И. (2005), Сравнителен анализ на използването на UML в приложения за реално време, "Развитие, бъдеще, перспективи и иновации в науката - 2005", стр.103÷113, София, 22 юни.
3. Batchkova I., DimitrovaD., **Antonova I.**(2005), Reliability Improvement of Control Systems using Formal Approaches, Материали от Международен симпозиум "Управление на топлоенергийни обекти и системи", стр.43÷46, Бобовдол, 3-4 ноември.

4. **Антонова И**., Бачкова И. (2006), Обектно-ориентирано проектиране на системи за управление на базата на UML и CORFU-FBDK, сборник доклади на научна сесия "Развитие - бъдещи перспективи и иновации в науката 2006", стр. 5÷16, 12 декември, София.

5. **Antonova I.**, Batchkova I. (2007), Design of Distributed Control Systems Using UML/SysML, International Conference Automatics and Informatics'07, pp. V-21÷V–24, 3-6 October 2007, Sofia, Bulgaria.

6. **Антонова И**., Бачкова И. (2007), Проектиране на разпределени системи за управление на базата на UML/SysML и стандарта IEC61499, International Scientific Conference, AMTEX'07, pp. ||-41÷||-47, 23-24 November, Gabrovo, Bulgaria.

7. **Antonova I.**, Batchkova I. (2008), Development of Multi-Agent Control Systems using UML/SysML, In Proceedings of the 4-th International IEEE Conference on Intelligent Systems, Vol.1, pp.6-26÷6-31, September 6-8, Varna, Bulgaria.

8. Михайлова Д., **Антонова И**., Бачкова И. (2008), Анализ на автоматичното генериране на JAVA код за системи за реално време с използване на UML и Rational Rose, Материали на Международната конференция "Автоматика и Информатика", стр. V-19÷V-22, 01-04 октомври, София.

9. Иванова Д., **Антонова И**., Бачкова И. (2008), Разработка на системи за управление на жилищния комфорт с използване на UML/SYSML и стандарта IEC 61499, Материали от Международен симпозиум "Управление на топлоенергийни обекти и системи", стр.51÷54, 14-15 ноември, Старозагорски Минерални Бани.

10. **Antonova I.**, Batchkova I. (2008), Design and analysis of IEC61499 based distributed control systems using UML and Rhapsody, Материали от Национална научно-техническа конференция с международно участие „Автоматизация в минната индустрия и металургията" БУЛКАМК'08, стр.173÷178, 27-28 ноември, София.

11. **Антонова И.** (2009), Приложение на UML и SysML за моделиране на системи за управление, базирани на стандарта IEC 61499, Proceeding of the International Conference „Automatics and Informatics", pp. III-13÷III-16, 29.09-04.10, Sofia.

12. **Antonova I.** (2010), Application of Uml-Mast for Modeling of Control System, Proceeding of the International Conference „Automatics and Informatics", pp. III-435÷III-438, October 3 – 7, Sofia.

13. Batchkova I., **AntonovaI.**, Ivanova D. (2011), Approaches of Industrial Informatics for Development of Distributed Control System, Anniversary Scientific Conference „40 YEARS DEPARTMENT OF INDUSTRIAL AUTOMATION", University of Chemical Technology and Metallurgy, pp. 171÷175, 18 March 2011, Sofia.

14. **Antonova I.** (2011), Application of Object-Oriented Approach in Control Systems, Anniversary Scientific Conference „40 YEARS DEPARTMENT OF INDUSTRIAL AUTOMATION", University of Chemical Technology and Metallurgy, pp. 189÷192, 18 March 2011, Sofia.

15. Batchkova I., **Antonova I.** (2011), Improving the Software Development Life Cycle in Process Control using UML/SysML, Preprints of the 18th IFAC World Congress, pp. 14133÷14138, August 28 - September 2, Milano, Italy.

16. Бачкова И., **Антонова И.**, Теллалян Ж. (2011), Подходи на софтуерното инженерство за подобряване на разработката на IEC-61499 базирани разпределени системи за управление, Proceedings of International Conference Automatics and Informatics'11, ISBN 1313-1850, pp.B-423÷ B-428, 03.10-07.10.2011, Sofia, Bulgaria.

УЧАСТИЯ В КОНФЕРЕНЦИИ С ДОКЛАД С РЕЗЮМЕ

1. **Антонова И**., Информационно моделиране на системи в реално време с използване на UML, "ЕКСПО ИНТЕЛЕКТ 2004", София, 1-2 декември, 2004.
2. Иванова И., Гочева Д., **Антонова И**., Съвместно използване на функционални и онтологични модели за целите на стандартизирания обмен на данни в нефтопреработвателната промишленост, "ЕКСПО ИНТЕЛЕКТ 2004", София, 1-2 декември, 2004.
3. Бачкова И., **Антонова И.**, Обектно-ориентирано проектиране на децентрализирани системи за управление с използване на UML-MAST, Научна конференция с международно участие " 60 години катедра Неорганична химия", 11.11.2005, София.

УЧАСТИЯ В КОНФЕРЕНЦИИ С ПОСТЕР

1. **Антонова И**., Бачкова И., Разработка на мулти-агентни системи с използване на UML/SysML, сборник с резюмета „Науката за устойчиво развитие по време на икономическа криза" от VI-та научна постерна сесия за студенти, докторанти и млади учени, София, 2009, стр. 67.
2. Бачкова И., **Антонова И**., Иванова Д., Подходи на индустриалната информатика за разработка на разпределени системи за управление, от VIII научна постерна сесия за студенти, докторанти и млади учени, София, 2011, стр. 139.
3. Batchkova I., **Antonova I**., Improving the Software Development Life Cycle in Process Control using UML/SysML, IX научна постерна сесия, София, 2012, стр. 154.

ЗАБЕЛЯЗАНИ ЦИТАТИ

Antonova I., Batchkova I. (2008), Development of Multi-Agent Control Systems using UML/SysML, In Proceedings of the 4th International IEEE Conference on Intelligent Systems, Vol.1, pp.6-26: 6-31, September 6-8,Varna, Bulgaria.

1. **Thramboulidis K., BudaA., 3+1 SysML view model for IEC61499 Function Block control systems Industrial Informatics (INDIN), 2010 8th IEEE International Conference, pp. 175 – 180**
2. **Zhenghui Sha, Qize Le and Jitesh H. Panchal*, USING SYSML FOR CONCEPTUAL REPRESENTATION OF AGENT-BASED MODELS, Proceedings of the ASME 2011 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE 2011 August 28-31, 2011, Washington, DC, USA**

Batchkova, I., Popov G., Stambolov, G., Antonova I. (2006). Design of open distributed control systems using IEC 61499, Journal of the TU Plovdiv on Fundamental Sciences and Applications., Vol.13, pp.45-51.

3. **Х. Карамишев, IEC-61499 базирано управление на подавателни движения в металорежещи машини, International Conference Automatics and Informatics'10, 3 - 7 October 2010, Sofia, Bulgaria, pp.III-427 – III-430.**

Антонова И., Бачкова И. (2007), Проектиране на разпределени системи за управление на базата на UML/SysML и стандарта IEC61499, International Scientific Conference, AMTEX'07, pp. ||-41÷||-47, 23-24 November, Gabrovo, Bulgaria

4. **Popov G., Geshev T., Stambolov G., Karamishev H., IEC61499 базирано управление на работна станция FESTO S-BE-M, Proceeding of the International Conference „Automatics and Informatics", October 3 – 7, pp. III-447 ÷ III-450**