ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – СОФИЯ

Машиностроителен факултет

Катедра Автоматизация на дискретното производство

маг. инж. Борян Чавдаров Владимиров

ПРОГРАМИРАНЕ НА ПРОМИШЛЕНИ РОБОТИ С ИЗПОЛЗВАНЕ НА API

А В Т О Р Е Ф Е Р А Т

на дисертация за придобиване на образователна и научна степен
"ДОКТОР"

Област: 5. Технически науки

Професионално направление: 5.1 Машинно инженерство

Научна специалност: Автоматизация на производството.

Научен ръководител: проф. д-р Стилиян Николов

СОФИЯ, 2025 г.

Дисертационният труд е обсъден и насочен за защита от Катедрения съвет на катедра „Автоматизация на дискретното производство“ към Машиностроителен факултет на ТУ-София на редовно заседание, проведено на 10.11.2025 г..

Публичната защита на дисертационния труд ще се състои на 29.01.2025 г. от 13.00 часа в Конферентната зала на БИЦ на Технически университет – София на открито заседание на научното жури, определено със заповед № ОЖ-5.1-104 / 04.12.2025 г. на Ректора на ТУ-София в състав:

1. проф. д-р Панчо Томов – председател
2. доц. д-р Велизар Захаринов – научен секретар
3. проф. д-р Младен Милушев
4. проф. д-р Николай Стоименов
5. доц. д-р Иванка Пеева

Рецензенти:

1. доц. д-р Велизар Захаринов
2. доц. д-р Иванка Пеева

Материалите по защитата са на разположение на интересуващите се в канцеларията на Машиностроителен факултет на ТУ-София, блок № 4, кабинет № 3242.

Дисертантът е задочен докторант към катедра „Автоматизация на дискретното производство“ на Машиностроителен факултет. Изследванията по дисертационната разработка са направени от автора, като някои от тях са подкрепени от научноизследователски проекти.

Автор: маг. инж. Борян Владимиров
Заглавие: Програмиране на промишлени роботи с използване на API

Тираж: 30 броя
Отпечатано в ИПК на Технически университет – София

I. ОБЩА ХАРАКТЕРИСТИКА НА ДИСЕРТАЦИОННИЯ ТРУД

Актуалност на проблема

Съвременните промишлени роботи (ПР), са високо енергийно ефективни, с възможност за изпълнение на широк кръг задачи. Това прави все по-привлекателно и икономически изгодно тяхното използване в различни области на индустрията, осигуряващо автоматизиране на различни производствени дейности и безопасна работна среда.

Развитието на компютърните технологии предостави възможност, чрез използването на Application Programming Interface (API), за създаване на потребителски приложения, които да автоматизират разработването на управляващи програми и техния трансфер към използваните ПР.

Всичко това, наред с необходимостта от високо квалифициран персонал за програмирането на ПР, прави актуални въпросите свързани с изследването на възможностите за автоматизирано и отдалечено програмиране на ПР.

Цел на дисертационния труд, основни задачи и методи за изследване

Цел на настоящата дисертационна работа е изследване на възможностите предоставяни от различни компютърни системи при разработването на УП за ПР, техния трансфер към ПР и възможностите за използване на API при работата на тези системи.

За изпълнението на тази цел се предвижда решаването на следните задачи:

1. Анализ на методите за програмиране на ПР
2. Анализ на възможностите на различни компютърни системи при разработването на УП за програмиране на ПР
3. Разработване на методика за разработването на УП за ПР с използване на компютърни системи
4. Изследване на възможностите, които API предоставя за програмиране на ПР с използване на компютърни системи
5. Анализ на методите и средствата за трансфер на данни при отдалечено програмиране на ПР
6. Разработване на API приложение и система за трансфер на данни при отдалечено програмиране на ПР
7. Прилагане на разработените приложения и оценка на тяхната функционалност

Изследванията са проведени в дигитална среда чрез прилагане на компютърни и комуникационни технологии.

Научна новост

- Разработена е универсална методика, за генериране на управляващи програми за ПР с използване на CARC системи.
- Определени са възможностите за използване на API при програмиране и експлоатация на ПР.
- Разработена е класификация на хакерски атаки срещу роботизирани производствени системи, проблемите които те предизвикват и са дадени препоръки за защита от тези атаки.

Практическа приложимост

С използване на API са разработени: сървърен и клиентски код за TCP/IP свързване; сървър написан на KAREL и код за криптиране на данни при експлоатация на ПР. Дефинирани са основните стъпки при проектиране на роботизирани клетки с използване на CARC системи.

Получените резултати се използват при обучението на студенти в катедра „Автоматизация на дискретното производство“.

Апробация

Апробацията на резултатите е проведени с използване на наличните в катедра „Автоматизация на дискретното производство“ ПР и софтуер за неговото програмиране.

Публикации

Основните постижения и резултати от дисертационния труд са публикувани в 6 на брой научни публикации, една от които е самостоятелна, а останалите са в съавторство с научният ръководител и колеги от катедра „Автоматизация на дискретното производство“.

Четири от публикациите са представени в България, на МНТК „Автоматизация на дискретното производство“, а две на научни форуми в чужбина.

Две от публикациите са индексирани в Scopus.

Структура и обем на дисертационния труд

Дисертационният труд е в обем от **193** страници, като включва увод, **7** глави за решаване на формулираните основни задачи, списък на основните приноси, списък на публикациите по дисертацията и използвана литература. Цитирани са общо **138** литературни източници, като **58** са на латиница и **7** на кирилица, а останалите са интернет адреси. Работата включва общо **89** фигури и **13** таблици. Номерата на фигурите и таблиците в автореферата съответстват на тези в дисертационния труд.

II. СЪДЪРЖАНИЕ НА ДИСЕРТАЦИОННИЯ ТРУД

ГЛАВА 1 ОБЗОР И АНАЛИЗ

1.1. История на развитие на промишлените роботи

Използването на роботи в индустрията, придобива първостепенно значение през миналия век. Разгледани са етапите на еволюцията на ПР.

1.2. Видове ПР и области на тяхното приложение

Въпреки че ПР са с различен дизайн, в зависимост от задачите които ще изпълняват, най-често срещани са т.нар. автоматизирани ръце „arms“. Тези роботи могат да бъдат класифицирани в няколко различни категории въз основа на движение, приложение, архитектура и модел и др.

Разгледани са основните типове ПР, според Международната федерация на роботите [33].

1.3. Системи за управление на ПР

Разгледани са градивните елементи на системи за управление, използвани в роботи и роботизирани устройства, работещи както в промишлени така и в неиндустриални среди, според ISO 8373:2012 (ревизирано в ISO 8373:2021 [38]).

1.4. Програмиране на ПР

Програмирането представлява дефиниране на желаните действия и движения на ПР, така че той да ги изпълнява без намесата на оператор.

Интерпретирането на УП в ПР се извършва от контролерът. Той е тази част от ПР, която осъществява връзката между оператора/програмиста и останалите елементи (външни и вътрешни сензори, периферно оборудване и др.). Разгледани са основните подходи, използвани при програмиране на ПР.

1.5. Автоматизиране на процеса на генериране на управляващи програми с използване на API

Приложният програмен интерфейс API (Application Programming Interface) е инструмент, който осигурява интеграция между различни приложения, като предоставя на потребителя възможност за използване на език за програмиране в рамките на друго приложение.

Такива приложения могат да автоматизират определени общи или повтарящи се задачи, които потребителите изпълняват. Други приложения могат да комбинират команди по такъв начин, че да осигурят допълнителна функционалност, съобразена по-специално с нуждите на определени компании и индустрии.

Системите за компютърно подпомагане CAx (Computer Aided Technologies) на инженерния труд, предлагат на потребителите различни възможности за използване на API, които могат да се разделят в две групи.

1.6. Изводи

ПР играят ключова роля в автоматизацията на индустриалните процеси. Те извършват множество операции: заваряване, боравене с материали, монтаж, боядисване и много други, като сложността на изпълняваните задачи непрекъснато се увеличава.

Използването на ПР трансформира повечето индустриални системи. Тази трансформация е повлияна от:

- Внедряване на Интернет на нещата - все повече и повече, роботите ще използват интелигентни сензори за събиране на информация и за подобряване на ефективността на процесите.
- Използване на анализ на големи данни - данните, събрани от ПР, трябва да бъдат организирани в системи, които да посмаляват тяхното анализиране и получаване на отчети за, състоянието им.
- Използване на виртуални решения - виртуалните решения правят възможно анализирането на роботизирани системи, без да се налага прекъсване на производството.
- Използване на отворени архитектури за автоматизация - необходимо е координация между производителите за създаване на стандарти и документация, които улесняват въвеждането на ПР в реална производствена среда.

Възможността за лесно и ефективно програмиране на ПР е критична за тяхното използване и оптимизацията на производствените процеси.

Използването на различни компютърни системи и софтуерни платформи, които позволяват програмиране, симулация и управление на ПР значително ускорява процеса на управление на ПР и намалява вероятността от грешки.

Въпросите свързани с използването на различни компютърни системи при управление на ПР, са все по-актуални във връзка с концепцията на Industry 4.0.

Въз основа на направения обзор и анализ е дефинирана целта на настоящата дисертационна работа, изследване на възможностите предоставяни от различни компютърни системи за използване на API при експлоатацията на ПР.

ГЛАВА 2 ЕЗИЦИ ЗА ПРОГРАМИРАНЕ ОТ ВИСОКО НИВО ИЗПОЛЗВАНИ ПРИ АВТОМАТИЗИРАНО ПРОГРАМИРАНЕ НА ПР

2.1. Програмен език FANUC KAREL

Fanuc KAREL [43] е програмен език, разработен през 80-те години от GMF (General Motors Fanuc) Robotics - съвместно предприятие между японската компания Fanuc и американския автомобилен гигант GM (General Motors). GMF Robotics е една от първите компании, които започват масово производство на ПР за автомобилната индустрия.

2.2. Структура на езика KAREL

KAREL включва структури и модели, общи за езиците от високо ниво, както и функции, разработени специално за управление на ПР Fanuc. Той разполага със строго типизирани променливи, константи, персонализирани типове, процедури, функции и дава достъп до различни вградени функции, които може да не са достъпни през TP (Teach Pendant).

2.3. Програмен език ABB RAPID

Езикът RAPID [44] е създаден от ABB Robotics в началото на 90-те години като част от разработката на ново поколение ПР. Той е предназначен да направи програмирането на ПР по-лесно, интуитивно и мощно, като осигури по-добър контрол върху движенията, входовете/изходите и обработката на грешки. RAPID е въведен заедно с контролера S4, а в следващите десетилетия получава подобрения и разширения, особено с контролерите IRC5, които добавят функции като мрежова комуникация, IoT интеграция и AI възможности.

Табл.2.3 Характеристики на FANUC KAREL и ABB RAPID

Характеристика	Fanuc KAREL	ABB RAPID
Платформа	Промишлени роботи Fanuc	Промишлени роботи ABB
Тип на езика	Процедурен, силно типизиран, подобен на Pascal/Ada	Процедурен, опростен
Формат на програмата	PROGRAM name ... END name;	MODULE name ... ENDMODULE
Обявяване на променливи	VAR speed : INTEGER;	VAR num speed;
Тип данни	INTEGER, REAL, BOOLEAN, STRING[n], ARRAY, XYZWPR, TRANS	num, bool, string, array, robtarget, jointtarget, pose
Условни оператори	IF ... THEN ... ELSE ... ENDIF;	IF ... THEN ... ELSE ... ENDIF
Цикли	FOR ... ENDFOR; WHILE ... ENDWHILE;	FOR ... ENDFOR; WHILE ... ENDWHILE;
Функции и процедури	FUNCTION name : type ... END name; ROUTINE name ... END name;	FUNC type name ... ENDFUNC; PROC name ... ENDPROC;
Движение на робота	Чрез TP програми или позиционни регистри GET_POS_REG, SET_POS_REG	MoveJ p1, v100, z10, tool1; MoveL p2, v50, fine, tool1;
Входове/Изходи (I/O)	Достъп чрез вградени системни процедури (IO_STATUS, SET_PORT_VAL)	SetDO do1, 1; WaitDI di1, 1;
Обработка на грешки	BEGIN ... ERROR ... END	TRAP ... ENDTRAP ERROR ... ENDERROR

Табл.2.4 Сравнение на програмните езици ST, ABB RAPID и Fanuc KAREL

Характеристика	ST (Structured Text)	ABB RAPID	FANUC KAREL
Платформа	PLC (Siemens, Beckhoff, Allen-Bradley, Schneider и др.)	Промишлени роботи ABB (IRC5 контролери)	Промишлени роботи FANUC (R-J3, R-30iA, R-30iB контролери)
Тип на езика	Високо ниво, стандартизиран (IEC 61131-3), подобен на Pascal/Modula/C	Специализиран език за ПР, процедурен	Специализиран език за ПР, процедурен, подобен на Ada/Pascal
Синтаксис	Подобен на Pascal и C; стандартен за PLC	Опростен, интуитивен за роботни движения	Силно структуриран, подобен на Ada/Pascal
Леснота на научаване	Средна – изисква PLC познания	Висока – лесен за оператори и програмисти	По-сложен – изисква по-задълбочени познания
Гъвкавост	Универсален – управление на процеси, логика, комуникации	Средна – силен за роботи, ограничен извън тази област	Ниска – работи само с FANUC контролери
Използване	PLC програмиране: логически операции, управление на машини, мрежова комуникация	Програмиране на ABB ПР: движения, логика, сензори, интеграция	Програмиране на FANUC ПР: логика, сензори, разширени функции, системен контрол
Движения на роботи	Чрез външни библиотеки, ограничена поддръжка	Вградено и мощно управление на движения: MoveJ, MoveL, MoveC	Индиректно чрез позиционни регистри (GET_POS_REG, SET_POS_REG) или TP програми
Програмна структура	Функции, процедури, цикли	Модули, процедури (PROC/FUNC), обекти	Програми, подпрограми, ROUTINE/FUNCTION, глобални и локални променливи
Приложения	Автоматизация на машини, процеси, SCADA интеграция	Роботизирани приложения: заваряване, монтаж, палетизиране, CNC, обработка vision, мрежи	Разширено програмиране на ПР: логика, сензори, vision, мрежи
Обработка на грешки	TRY...CATCH, стандартен механизъм за изключения	ERROR и TRAP блокове за обработка	BEGIN ... ERROR ... END и ON ERROR конструкции
Поддръжка на мрежи	Modbus, EtherNet/IP, PROFINET и други стандартни PLC протоколи	TCP, UDP, FTP, Socket комуникация	TCP, FTP (няма UDP); ограничена мрежова интеграция

2.4. Структура на езика RAPID

Езикът RAPID за програмиране има подобен вид на повечето ST езици за програмиране и много наподобява езика C. Той е структуриран и лесен за научаване, като предоставя богат набор от инструкции и функции за

управление на движенията на ПР, обработка на сигнали, комуникация с външни устройства и много други.

2.5. Сравнителен анализ на възможностите на езици за програмиране от високо ниво използвани при автоматизирано програмиране на ПР

Fanuc KAREL и ABB RAPID са два различни програмни езика, използвани за управление на ПР. Въпреки че и двата езика имат подобни цели, те се различават по структура, синтаксис и предназначение. Сравнение на характеристиките на двата езика е даден в Табл.2.3.

Въпреки спецификата на KAREL и RAPID, всеки запознат с език за програмиране в стил C, би трябвало сравнително лесно да се справи с програмиране на ПР с използване на тези езици.

В Табл.2.4 е дадено сравнение на програмните езици ST, ABB RAPID и Fanuc KAREL, използвани в индустриалната автоматизация.

2.6. FANUC KAREL и ABB RAPID в контекста на Industry 4.0

Програмните езици Fanuc KAREL и ABB RAPID, предоставят допълнителни възможности при управлението на ПР, съвместими със съвременните концепции на Industry 4.0, като интернет на нещата IoT (Internet of Things), облачни технологии CT (Cloud Technologies), изкуствен интелект AI (Artificial Intelligence) и интелигентна автоматизация.

2.7. Изводи

Езиците за програмиране ST, ABB RAPID и Fanuc KAREL, използвани в индустриалната автоматизация, притежават някои общи черти, но всеки от тях е разработен за специфична платформа и има своите уникални особености.

Анализирани са допълнителните възможности, които езиците за програмиране KAREL и RAPID, предоставят на потребителите при програмиране на ПР.

Направен е сравнителен анализ на възможностите на езици за програмиране ST, ABB RAPID и Fanuc KAREL при автоматизирано програмиране на ПР.

Анализирано е използването на програмните езици Fanuc KAREL и ABB RAPID, в контекста на използването на: Интернет на нещата; облачни технологии; изкуствен интелект и интелигентна автоматизация при експлоатация на ПР.

ГЛАВА 3 ТРАНСФЕР НА ДАННИ ПРИ ЕКСПЛОАТАЦИЯ НА ПР

3.1. Протоколи за трансфер на данни при експлоатация на ПР

В настоящия момент, се използват множество технически решения, чрез които се осъществява отдалечено програмиране и управление на производствено оборудване, изпълняващо дадена производствена програма.

Сравнение на мрежовите функции и протоколи, поддържани в разгледаните в **Глава 2**, езици за програмиране от високо ниво използвани при експлоатацията на ПР е дадено в Табл.3.1.

Табл.3.1 Мрежовите функции и протоколи в ST, FANUC KAREL и ABB RAPID

Функция / Протокол	ST (PLC)	FANUC KAREL	ABB RAPID	Забележки / ограничения
Поддържани мрежови протоколи	TCP/IP, UDP, Modbus TCP, OPC-UA, MQTT, ProfiNet, EtherNet/IP	TCP/IP, FTP, HTTP, Socket Messaging	TCP/IP, UDP*, FTP, HTTP, Web Services (REST/SOAP), Socket Messaging	*RAPID: UDP не е вграден; изисква външна конфигурация; KAREL няма UDP
Създаване на сокет (TCP/IP клиент/сървър)	SysSockCreate(), SysSockBind(), SysSockConnect()	Комуникация чрез конфигурирани тагове, MSG_DISCO и OPEN FILE	SocketCreate, SocketBind, SocketConnect	UDP сокети при RAPID/KAREL изискват външна конфигурация или скриптове
Изпращане на данни	SysSockSend()	Socket Send Messaging	SocketSend	
Получаване на данни	SysSockRecv()	Socket Receive Messaging	SocketReceive (има в ръководството, но не винаги се използва)	
Затваряне на сокет	SysSockClose()	затваряне чрез стандартни файлови операции	SocketClose	
FTP комуникация	SysFTPClientPut(), SysFTPClientGet()	CALL_PROG	Вграден FTP сървър, достъп чрез GetFTP	
HTTP/HTTPS поддръжка	SysWebClientRequest() (при някои PLC)	HTTP_REQ	HTTP и Web Services (REST/SOAP)	
UDP комуникация	SysSockSendTo(), SysSockRecvFrom()	Не поддържа	UDPSend, UDPReceive	KAREL няма UDP; RAPID – само чрез допълнителна конфигурация и скриптове
WebSockets	Ограничена поддръжка	Ограничена поддръжка	Възможно чрез Web Services	Няма универсална вградена WebSocket поддръжка
Индустриални протоколи	OPC UA, Modbus TCP, ProfiNet, EtherNet/IP, MQTT	OPC UA (с лиценз), Modbus TCP, EtherNet/IP, ProfiNet	OPC UA (вграден), Modbus TCP, ProfiNet, EtherNet/IP	KAREL: OPC UA изисква лиценз; RAPID – вграден, пълна поддръжка
Поддръжка на JSON/XML	Възможна с библиотеки (зависи от производителя)	Ръчна обработка на низове	Вграден парсинг за JSON/XML	KAREL: ръчна обработка; ST PLC – зависи от наличните библиотеки
Вграден Web Server	Поддържан в някои PLC	Ограничена поддръжка	Вграден Web Server за мониторинг и управление	KAREL: ограничен; RAPID – пълен, включително ABB Ability интерфейс
Cloud интеграция	MQTT, OPC-UA към облачни услуги	Изисква външни скриптове – FANUC FIELD SYSTEM	Поддържа ABB Ability Cloud	KAREL: трябва TP скрипт или FIELD System; ST PLC – зависи от библиотеките

Табл.3.2 Сървърен код за TCP/IP свързване

FANUC KAREL	ABB RAPID

PROGRAM tcp_server
%STACKSIZE = 4000
%NOLOCKGROUP
%NOPAUSE=ERROR+COMMAND+TPENABLE
%ENVIRONMENT uif
%ENVIRONMENT memo
%ENVIRONMENT flbt
%INCLUDE klevkeys
%ENVIRONMENT bynam
%INCLUDE klevkmsk
%ENVIRONMENT fdev
%INCLUDE klevccdf
%ENVIRONMENT kclop
%ENVIRONMENT sysdef
VAR
file_var : FILE
tmp_int : INTEGER
tmp_str1 : STRING[128]
tmp_int1 : INTEGER
tmp_str : STRING[128]
status : INTEGER
entry : INTEGER
BEGIN
SET_FILE_ATR(file_var, ATR_IA)
SET_VAR(entry, 192.168.1.3 '*SYSTEM*',
'$HOSTS_CFG [3]. $SERVER_PORT', 6008, status)
WRITE TPPROMPT('Connecting.', status, CR)
MSG_CONNECT('S3', status)
WRITE TPPROMPT('Connect Status = ', status, CR)
IF status = 0 THEN
WRITE TPPROMPT('Opening', CR)
FOR tmp_int1 = 1 TO 20 DO
OPEN FILE file_var ('rw', 'S3:')
status = IO_STATUS(file_var)
WRITE TPPROMPT(status, CR)
IF status = 0 THEN
FOR tmp_int = 1 TO 1000 DO
WRITE TPPROMPT ('Reading', CR)
BYTES_AHEAD (file_var, entry, status)
WRITE TPPROMPT (entry, status, CR)
READ file_var (tmp_str::10)
status = IO_STATUS(file_var)
WRITE TPPROMPT(status, CR)
ENDFOR
CLOSE FILE file_var
ENDIF
ENDFOR
WRITE TPPROMPT('Disconnecting..', CR)
MSG_DISCO('S3:', status)
WRITE TPPROMPT('Done.', CR)
ENDI
END tcp_server
	MODULE Module1
	VAR socketdev serverSocket;
	VAR socketdev clientSocket;
	VAR string data;
	PROC main()
	!Add your code here
	SocketCreate serversocket;
	SocketBind serverSocket, “127.0.0.1” ,
	5000;
	SocketListen serverSocket;
	SocketAccept serverSocket,
	clientSocket, \Time:= WAIT_MAX;
	
	SocketReceive clientSocket
	\Str:=data;
	SocketSend clientSocket \Str:=
	“received”;
	SocketClose clientSocket;
	SocketClose serverSocket;
	ERROR
	IF ERRNO=ERR_SOCK_TIMEOUT
	THEN
	RETRY;
	IF ERRNO=ERR_SOCK_TIMEOUT
	THEN
	RETRY;
	ELSEIF
	ERRNO=ERR_SOCK_CLOSED THEN
	SocketClose clientSocket;
	SocketClose serverSocket;
	SocketCreate serverSocket;
	SocketBind serverSocket, “127.0.0.1” ,
	5000 ;
	SocketListen serverSocket;
	SocketAccpet serverSocket,clientSocket,
	\Time:=WAIT_MAX;
	RETRY;
	ELSE
	stop;
	ENDIF
	ENDPROC
	ENDMODULE

В Табл.3.2 е дадено сравнения на сървърен код за TCP/IP свързване разработен с Fanuc KAREL и ABB RAPID.

Табл.3.3 Клиентски код за TCP/IP свързване

FANUC KAREL	ABB RAPID

PROGRAM tcp_client
%STACKSIZE = 4000
VAR
file_var : FILE
status : INTEGER
recv_str : STRING[128]
send_str : STRING[16] := "Hello from client"
bytes_ahead : INTEGER
i : INTEGER
BEGIN
MSG_CONNECT('S3', status)
WRITE TPPROMPT('Connect Status = ', status, CR)
IF status = 0 THEN
OPEN FILE file_var ('rw', 'S3:')
status = IO_STATUS(file_var)
IF status = 0 THEN
FOR i = 1 TO 10 DO
WRITE TPPROMPT('Sending data...', CR)
WRITE file_var(send_str)
status = IO_STATUS(file_var)
IF status <> 0 THEN
WRITE TPPROMPT('Error sending data', CR)
EXIT FOR
ENDIF
 	
BYTES_AHEAD(file_var, bytes_ahead, status)
IF bytes_ahead > 0 THEN
READ file_var(recv_str::bytes_ahead)
WRITE TPPROMPT('Received: ', recv_str, CR)
ELSE
WRITE TPPROMPT('No data available to read', CR)
ENDIF
ENDFOR
CLOSE FILE file_var
ELSE
WRITE TPPROMPT('Error opening socket file', CR)
ENDIF
 	
MSG_DISCO('S3', status)
WRITE TPPROMPT('Disconnected', CR)
ELSE
WRITE TPPROMPT('Connection failed', CR)
ENDIF
END tcp_client
	MODULE Module1
	VAR socketdev clientSocket;
	VAR string data;
	VAR string recvData;
	VAR num errStatus;
	PROC main()
	! Създаване на клиентски сокет
	SocketCreate clientSocket;
	
	! Опит за свързване към сървъра на 127.0.0.1:5000
	SocketConnect clientSocket, "127.0.0.1", 5000 \Timeout:=5000; ! Проверка за грешки при свързване
	IF ERRNO <> 0 THEN TPWrite "Connection failed with error: "; TPWrite ERRNO; RETURN; ENDIF „
	! Изпращане на съобщение към сървъра data := "Hello Server"; SocketSend clientSocket \Str:=data; ! Получаване на отговор от сървъра
	SocketReceive clientSocket \Str:=recvData;
	! Показване на получените данни
	TPWrite "Received from server: " & recvData;
	! Затваряне на клиентския сокет
	SocketClose clientSocket; ENDPROC
	ENDMODULE

При експлоатацията на ПР могат да се разграничат две взаимно свързвани групи задачи: генериране на УП и трансфера на данни между контролера на

робота и платформата на която е инсталиран софтуера за отдалечено програмиране и управление.

3.2. Свързване на ПР в мрежа

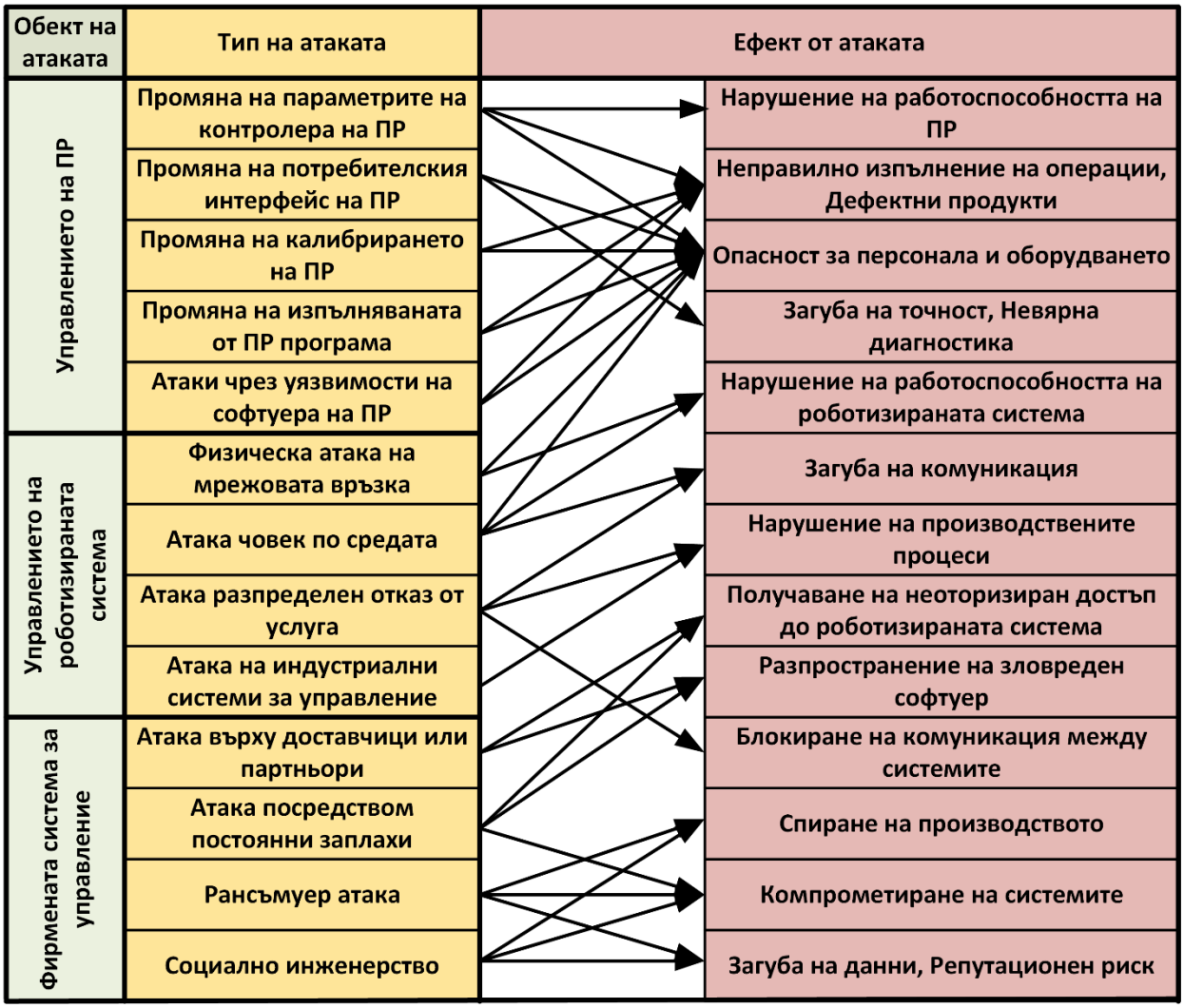
Свързването на ПР в мрежа позволява интеграцията му със съществуващи автоматизирани производствени системи, а също така предлага възможности за дистанционно управление, мониторинг и анализ на данни.

В Табл.3.3 е дадено сравнения на клиентски код за TCP/IP свързване разработен с Fanuc KAREL и ABB RAPID.

3.3. Мрежова сигурност при експлоатация на ПР

С увеличаване на сложността на индустриалните мрежи се увеличават и заплахите за тяхната сигурност, а проектиращите подобни технически решения, трябва да вземат в предвид всякакви видове заплахи, за да осигурят, сигурността на предаваната информация.

Отдалеченото програмиране на ПР се базира, на сигурна мрежова връзка чрез криптиращи информацията, мрежови устройства, обезпечаващи сигурността на предаваната информация.



Фиг.3.2 Класификация на атаките срещу роботизирани производствени системи

3.4. Атаки при експлоатация на ПР

На база извършен анализ на видовете атаки срещу роботизираните производствения системи е разработена тяхната класификация по три признака, показана на Фиг.3.2.

Табл.3.4 Видове атаки срещу роботизирани системи

Тип атака	Подходи за защита
Атаки срещу управлението на ПР	
Промяна на параметрите на контролера на ПР	Контрол на достъпа, засичане на аномалии, валидация на параметър, двуфакторна автентикация (2FA), преинсталация от backup
Промяна на потребителския интерфейс на ПР	Контрол на достъпа, мониторинг на данните, шифроване на комуникацията, редовни бекъпи
Промяна на калибрирането на ПР	Контрол на достъпа, проверка на калибрацията, засичане на аномалии, цифрови подписи, редовни бекъпи, прекалибриране
Промяна на изпълняваната от ПР програма	Контрол и ограничаване на достъпа, резервни копия на програмите, презареждане на програмата
Атаки чрез уязвимости на софтуера на ПР	Контрол на достъпа, провеждане на пен-тестове за откриване на уязвимости, редовни backup-и, редовни актуализации на софтуера.
Атаки срещу управлението на роботизираните производствени системи	
Физически атаки чрез мрежовата връзка	Контрол и ограничаване на достъпа до мрежовите портове, защитени зони, физическа защита на мрежата, VPN защити
Атаки човек по средата - Man-in-the-Middle (MITM)	Шифроване на мрежовия трафик, VPN допълнителна стена, ежедневен бекъп на роботизираната система, сегментиране на мрежата
Атаки разпределен отказ от услуга - Distributed Denial-of-Service (DDoS)	Наблюдение на мрежовия трафик, защитни стени, DDoS защита, мрежова сегментация
Атаки на индустриални системи за управление	Мониторинг и анализ на системата, сегментиране на мрежата, защита на ICS системи
Атаки срещу фирмената система за управление	
Атаки срещу доставчиците и партньорите - Supply Chain Attacks	Контрол на мрежата за приемане на данни, отдалечен мониторинг на мрежата, проверка на софтуера, цифрови подписи
Атаките посредством постоянни заплахи Advanced Persistent Threats	Мониторинг на мрежовия трафик, за да откривате подозрителни активности, обучение на екипа за идентифициране на продължителни атаки, редовни одити
Рансъмуер атаки	Мониторинг за откриване на зловреден код, редовни бекъпи на данни, антивирусна защита, мрежова сегментация
Социално инженерство	Политики за сигурност, тестване на сигурността, ограничаване на споделянето на чувствителна информация, верификация на потребителите, който изискват достъп

Тези признаци са:

- **Атаки срещу управлението на ПР**

Първата група са атаки насочени към контролера на ПР. Те имат за цел да променят поведението на ПР при, неговата експлоатация.

➢ **Атаки срещу управлението на роботизираните производствени системи**

Втората група са атаки насочени към мрежата, в която е свързан контролерът на ПР. Те имат за цел да проникнат в системата за управление на роботизираната система и да нарушат нейната работа.

➢ **Атаки срещу фирмената система за управление**

Третата група са атаки целящи получаване на достъп до роботизираните системи без необходимите права, с цел разпространение на зловреден софтуер, който да наруши работата на системите.

Ефектите от атаката са: получаване на неоторизиран допуск; спиране на производството; загуба на данни; финансови загуби; репутационен риск.

В Табл.3.4 са дадени подходите, които могат се използват за защита от разгледаните атаки.

3.5. Криптиране на данни при експлоатация на ПР

Предимството от използването на системите с отворен код е, че са свободни за доизграждане и технологично развитие, според потребностите на потребителите.

При използване на криптиращи устройства се залага на сигурността на предаваната информация, като по този начин се предотвратява възможността за загуба или неоторизиран достъп до чувствителна информация. В настоящият момент сигурността на индустриалните мрежи, не може да бъде пренебрегвана, в резултат на което на пазара се предлагат все повече решения, за криптиране на данни, на приемлива цена.

3.6. Изводи

Свързването на ПР в мрежа е ключово за съвременните индустриални приложения. Възможността за комуникация през мрежата не само подобрява производителността, но и позволява дистанционно управление и мониторинг на роботизираната система.

Разгледани са основните комуникационните протоколи, чрез които може да се осъществи отдалечена експлоатация на ПР и са анализирани мрежовите функции заложени в езиците за програмиране на ПР ST, Fanuc KAREL и ABB RAPID

Разработени и сравнени са сървърен и клиентски код за TCP/IP свързване с използване на езиците за програмиране на ПР Fanuc KAREL и ABB RAPID.

Разработена е класификация по три признака на, видовете атаки срещу роботизирани производствени системи и са дадени препоръки за защита от тези атаки.

Криптирането на данни при програмирането на ПР играе важна роля за осигуряване на безопасност и конфиденциалност на комуникацията между роботизираните системи и външните устройства. Това е особено важно, когато роботизираната система е интегрирана в индустриална мрежа, където обменът на чувствителни данни може да бъде изложен на атаки и неоторизиран достъп. Криптирането помага да се защитят данните и комуникационните канали,

което е критично за поддържането на конфиденциалност и цялост на системата.

Разработен е пример за защита на сокет комуникацията при експлоатация на ПР Fanuc, с използване на езика KAREL.

ГЛАВА 4 АВТОМАТИЗИРАНЕ ПРОГРАМИРАНЕТО НА ПР

4.1. Компютърни системи за програмиране на ПР

Това са CARC (**C**omputer **A**ided Robot **C**ontrol) [46] системите, които могат да се срещнат и като OLPE (**O**ff-**l**ine **P**rogramming **E**nvironments). В тях се прилагат съвременните достиженията в областта изчислителната техника и инженерната компютърна графика.

CARC системи използват 3D модели на ПР и друго производствено оборудване, чрез които могат да се изграждат роботизирани производствени системи, да се симулира и анализира тяхната работа и да се генерират управляващи програми, за включените в тях ПР.

4.2. Анализ на възможностите на основните съвременни CARC системи

Отчитайки, многообразието на предлагани в момента от различни производители CARC системи, използвайки общо достъпни източници е направено изследване на техните функционални възможности на база на следните десет критерия:

- наличие на специализирани модули за програмиране при изпълнение на различни задачи от ПР;
- наличие на библиотеки с 3D модели на различни ПР;
- наличие на библиотеки с 3D модели на КИЗ;
- наличие на библиотеки с 3D модели на производствено оборудване;
- възможност за използване на виртуален пулт за управление на ПР;
- възможност за използване на езици за програмиране на ПР от високо ниво и API;
- възможност за симулиране и оптимизиране на работата на ПР;
- възможност за проектиране и анализ на работата на роботизирани клетки;
- възможност за обмен на данни с други CAx системи;
- поддържане на различни контролери на ПР.

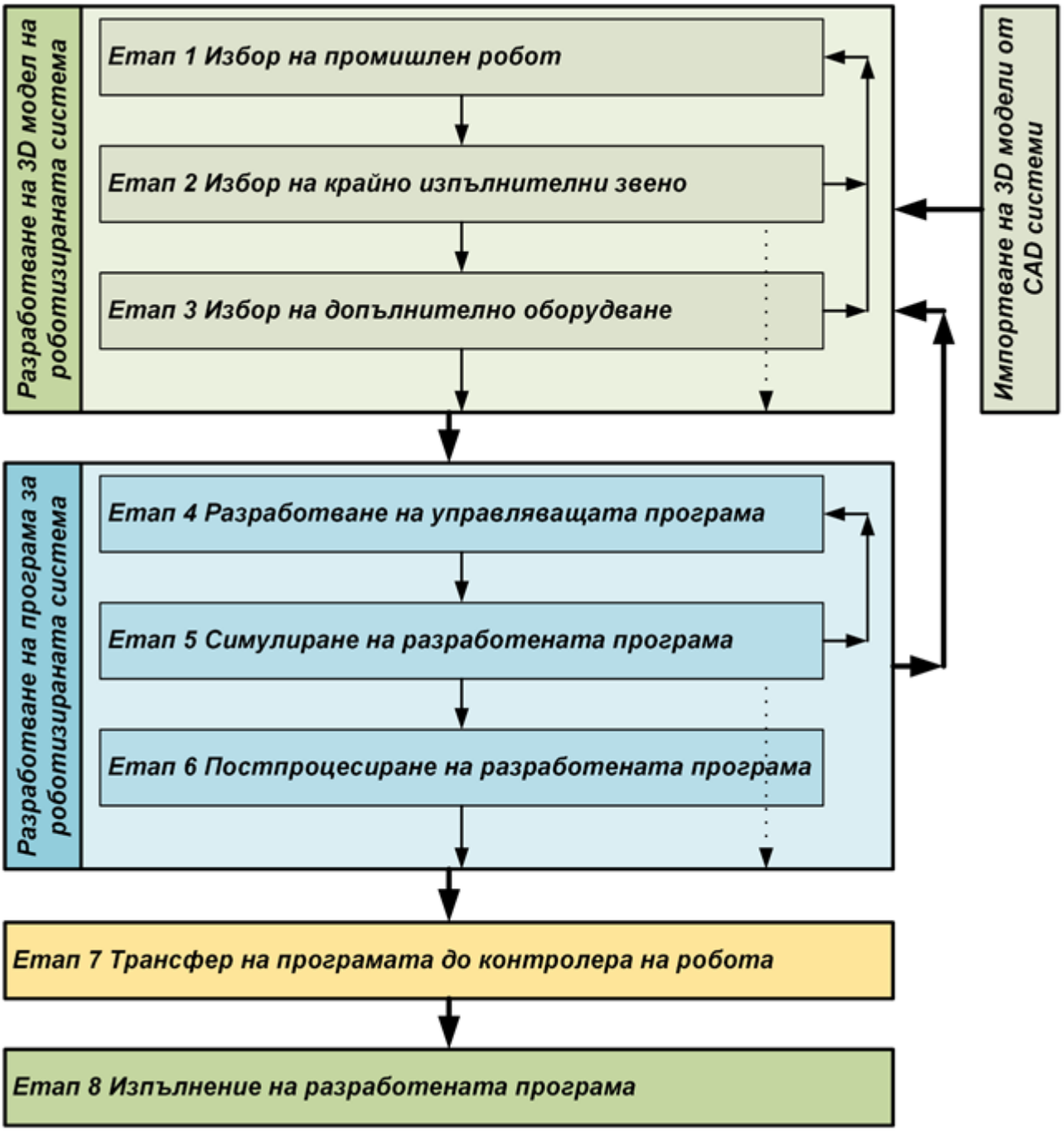
Резултатите от направеното изследване са дадени в **Приложение 1**. От данните там се вижда, че основните различия в предлаганите на пазара CARC системи, са свързани с възможностите им да обменят данни с други CAx системи и поддържане на контролери на ПР, предлагани от различни фирми.

4.3. Методика за генерирана е на управляващи програми за промишлени роботи с използване на CARC системи

Независимо от разнообразието на предлаганите в момента CARC системи, основните етапи при работа с тези системи, до голяма степен се припокриват. Тук е предложена методика описваща основните стъпки при работа с CARC

системите, необходими за генериране на УП за ПР и са анализирани възможностите за използване на API в този процес.

Основните етапи, необходими за генериране на УП за ПР, независимо от използваната CARC система са дадени в разработената методика показана на Фиг.4.1.



Фиг.4.1 Методика за генерирана е на УП за ПР с използване на CARC системи

Те са както следва:

Етап 1 Избор на промишлен робот

Етап 2 Избор на крайно изпълнителни звено

Етап 3 Избор на допълнително оборудване

Етап 4 Разработване на управляващата програма

Етап 5 Симулиране на разработената програма
Етап 6 Постпроцесиране на разработената програма
Етап 7 Трансфер на програмата до контролера на робота
Етап 8 Изпълнение на разработената програма

4.4. Изводи

Направен е сравнителен анализ на функционалните възможности на различни съвременни CARC системи предлагани на пазара, на база на десет критерия.

Съвременните CARC системи, предоставят на потребителите широки възможности за генериране УП за ПР, те позволяват в процеса на разработване на УП, да се използват различни 3D модели, виртуални пултове, API и езици за програмиране от високо ниво.

Разработена е методика, включваща осем етапа, за генерирана е на УП за ПР с използване на CARC системи.

Разгледани са основните стъпки при използване на разработената методика и са анализирани възможностите за използване на API в този процес.

Разработената методика, за генерирана е на УП за ПР с използване на CARC системи е универсална и не зависи от използваната CARC система и ПР, за които се генерира УП.

ГЛАВА 5 ИЗПОЛЗВАНЕ НА API ПРИ ЕКСПЛОАТАЦИЯ НА ПР

5.1. API в компютърните системи за програмиране на ПР

Отчитайки отделните стъпки в предложената методика и направения анализ на възможностите, който съвременните CARC системи, предоставят на потребителите за генериране УП за ПР използването на API в този процес може да бъде използвано за:

- разработването на софтуерни приложения, за създаване на 3D модели на различни компоненти за проектиране на роботизирани клетки;
- разработване на приложения за отдалечено програмиране на ПР;
- разработване на приложения за отдалечен обмен на данни между ПР и хост компютър.

API предоставя интерфейс, който позволява на различни софтуерни приложения да комуникират и взаимодействат едно с друго. В контекста на експлоатацията на ПР, API играе ключова роля в следните аспекти:

5.2. Езици за програмиране използвани в API при експлоатация на ПР

В Табл.5.1 е дадено сравнение на някои основни характеристики при използване на езиците Fanuc KAREL, ABB RAPID, C++, C#, Python и Perl, при експлоатация на ПР.

Табл.5.1 Сравнение на езиците Fanuc KAREL, ABB RAPID, C++, C#, Python и Perl, при експлоатация на ПР

Функция / Описание	Fanuc KAREL	ABB RAPID	C/C++	C#	Python	Perl
Създаване на сокет	MSG_CONN / MSG_CONNECT	SocketCreate socketdev	socket()	TcpListener<	socket.socket()	IO::Socket::INET->new(...)
Свързване към сървър	MSG_DISCO	SocketConnect socketdev, ip, port	connect()	TcpClient	socket.connect()	connect()
Изпращане на данни	Команди на KAREL	SocketSend socketdev, data	send()	NetworkStream.Write()	send()	print SOCKET "data"
Получаване на данни	WRITE	SocketReceive socketdev, buffer	recv()	NetworkStream.Read()	recv()	$data = <SOCKET>
Затваряне на сокет	MSG_DISCO(tag_name, status)	SocketClose socketdev	close()	Close()	close()	close(SOCKET)
Поддържани протоколи	TCP/IP	TCP/IP, UDP	TCP/IP, UDP	TCP/IP, UDP	TCP/IP, UDP	TCP/IP, UDP
Мулти клиентска поддръжка	Ограничена сървър	Да	Да, чрез многопоточност (pthread/std::thread)	Да, чрез Task/Threading	Да, чрез threading или asyncio	Да, чрез fork()
Обработка на грешки	BEGIN ... ERROR Обработва грешки чрез ERROR блокове	TRAP Обработва грешки чрез TRAP блокове	try-catch Обработва грешки чрез try-catch блокове.	try-catch Обработва грешки чрез try-catch блокове.	try-except Обработва грешки чрез try-except блокове.	eval и $@ Обработва грешки чрез eval и $@.
Приложения в индустриална автоматизация	Робот-контролери, комуникация с PLC	SCADA, IoT, Web интеграции	Вградени системи, комуникация с роботи	Индустриални приложения, SCADA	IoT, AI, обработка на данни, роботи	Уеб услуги, автоматизация
Функция / Описание	FANUC KAREL	ABB RAPID	C/C++	C#	Python	Perl
Създаване на сокет	MSG_CONN(tag_name, status)	SocketCreate socketdev	socket()	TcpListener	socket.socket()	IO::Socket::INET->new(...)

В Табл.5.2 е дадено сравнения на клиентски код за TCP/IP свързване разработен с езици за програмиране от високо ниво C++, C#, Python и Perl. Клиентски код за TCP/IP при експлоатация на ПР, разработен с езиците Fanuc KAREL, ABB RAPID е даден в Табл.3.3.

В Табл.5.3 е дадено сравнение на използването на различни езици за програмиране от високо ниво, използвани за криптиране на данни при експлоатация на ПР.

Табл.5.2 Сравнение клиентски код за TCP/IP свързване

Език от високо ниво	TCP клиент – сорс код
Python	import socket client = socket.socket(socket.AF_INET, socket.SOCK_STREAM) client.connect(("192.168.1.100", 5000)) client.send(b"Hello, Server!") response = client.recv(1024) print("Received:", response.decode()) client.close()
PERL	use IO::Socket::INET; my $client = IO::Socket::INET->new(PeerHost => "192.168.1.100", PeerPort => "5000", Proto => "tcp") or die "Could not connect\n"; print $client "Hello, Server!\n"; my $response = <$client>; print "Received: $response\n"; close($client)

C++	#include <iostream>
	#include <sys/socket.h>
	#include <arpa/inet.h>
	#include <unistd.h>
	int main() {
	int sock = socket(AF_INET, SOCK_STREAM, 0);
	struct sockaddr_in server;
	server.sin_family = AF_INET;
	server.sin_port = htons(5000);
	inet_pton(AF_INET, "192.168.1.100", &server.sin_addr);
	connect(sock, (struct sockaddr*)&server, sizeof(server));
	send(sock, "Hello, Server!", 14, 0);
	char buffer[1024] = {0};
	recv(sock, buffer, 1024, 0);
	close(sock);
	return 0;
	}

C#	using System;
	using System.Net.Sockets;
	using System.Text;
	class Program {
	static void Main() {
	TcpClient client = new TcpClient("192.168.1.100", 5000);
	NetworkStream stream = client.GetStream();
	byte[] data = Encoding.ASCII.GetBytes("Hello, Server!");
	stream.Write(data, 0, data.Length);
	byte[] buffer = new byte[1024];
	stream.Read(buffer, 0, buffer.Length);
	Console.WriteLine("Received: " + Encoding.ASCII.GetString(buffer));
	client.Close();
	}
	}

5.3. Приложения на API в индустриалната автоматизация

Използването на API в индустриалната автоматизация, може да се разглежда в следните аспекти:

- автоматизация на производството - позволява синхронизация на работата на ПР и друго производствено оборудване;
- колаборативни роботи (Cobots) - използва се за управление на роботи, които работят съвместно с хора;
- интелигентни фабрики - осигурява интеграция на ПР със системи за IoT и машинно обучение.

5.4. Изводи

API в компютърните системи за програмиране на ПР, предоставя мощен инструмент за интеграция, автоматизация и управление. Той улеснява разработката на софтуерни решения и гарантира, че ПР могат да бъдат адаптирани към динамично променящите се изисквания на съвременната индустрия.

Разгледани са възможностите и аспектите, които API предоставя при генериране УП за ПР и тяхната експлоатация.

Разгледани са най-често използваните езици за програмиране от високо ниво, използвани в API при експлоатация на ПР и са анализирани областите на тяхното приложение.

Разработен и анализиран е код, за криптиране на данни при експлоатация на ПР, с използване на различни езици за програмиране от високо ниво, използвани в API при експлоатация на ПР.

Направено е сравнение на някои от основни характеристики при използване на езиците Fanuc KAREL, ABB RAPID, C++, C#, Python и Perl, при експлоатация на ПР.

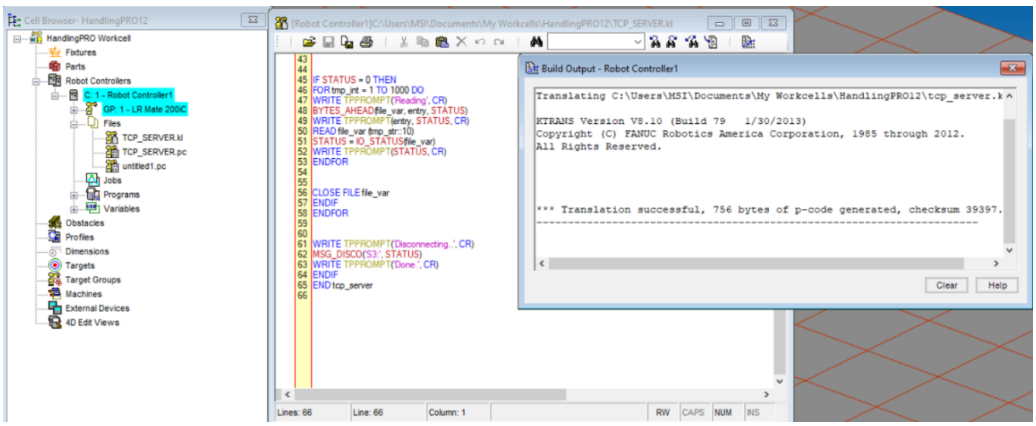
Създадена е клиент сървър система като са използвани Common LISP и Emacs LISP.

ГЛАВА 6 ПРИЛАГАНЕ НА РАЗРАБОТЕНАТА МЕТОДИКА ЗА ПРОГРАМИРАНЕ НА ПР

6.1. Програмиране на ПР в средата на ROBOGUIDE

Разработената методика е използвана за програмиране на ПР в средата на ROBOGUIDE.

С използване на Fanuc ROBOGUIDE, създадения сървър TCP_SERVER.kl написан на KAREL е компилиран Фиг.6.9.

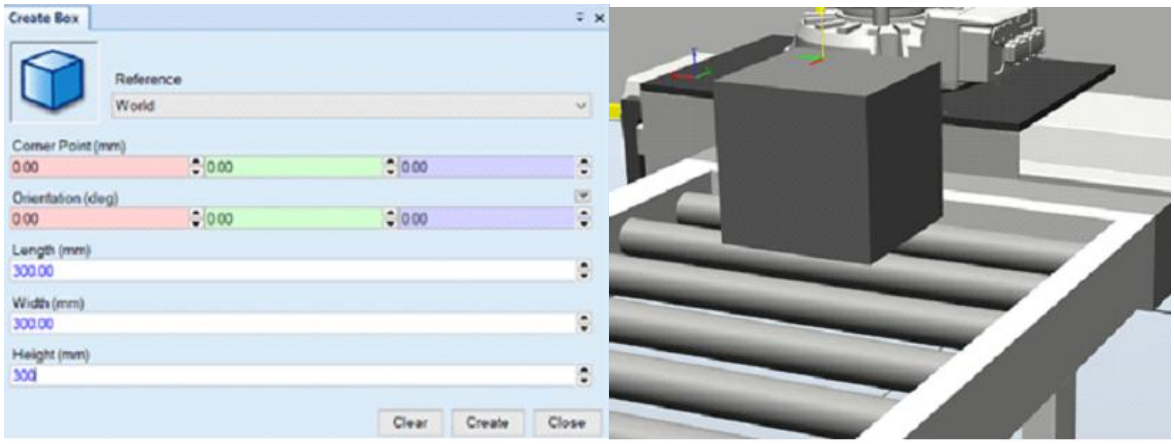


Фиг.6.9 Компилиране на TCP_SERVER.kl във Fanuc ROBOGUIDE

6.2. Програмиране на ПР в средата на ABB RobotStudio

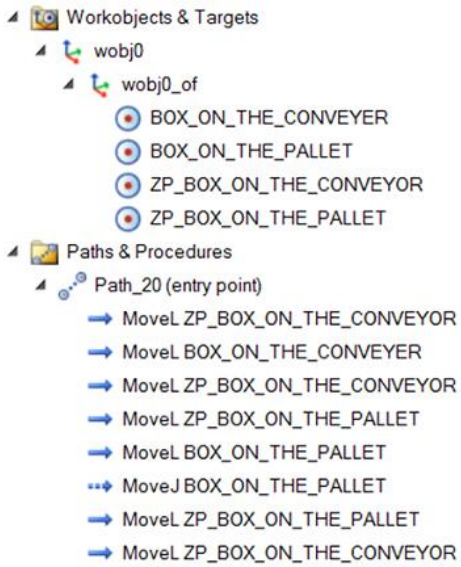
Разработената методика е използвана за програмиране на ПР в средата на ABB Robot Studio.

Манипулирания детайл е кашон /box/ с желаните размери и тегло Фиг.6.23.



Фиг.6.23 Дефиниране на манипулирания обект

Следващата стъпка е дефиниране на движенията на ПР между дефинираните точки Фиг.6.27.

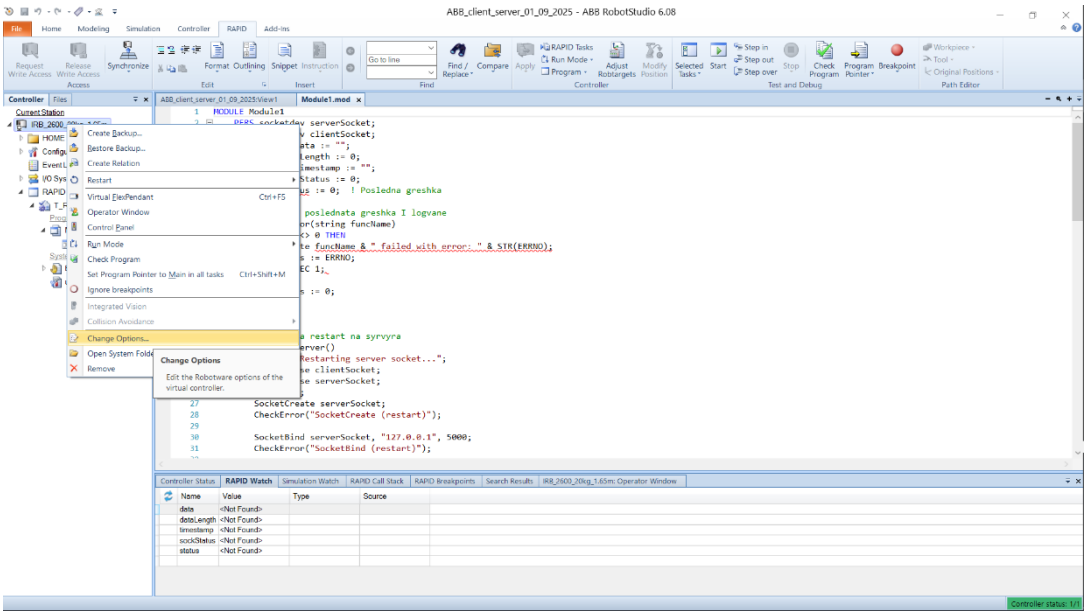


Фиг.6.27 Дефиниране на движенията на ПР

Разработената програма може да прехвърли данни тип “String” – низове от външен клиент написан на език на високо ниво, до сървъра на ABB RAPID. Софтуерната опция, инсталирана на ABB RobotStudio е “616-1 PC Interface” – необходима за Socket комуникация.

Предаване на данни е показано на Фиг.6.30:

- data “string”/ред 1 - VALUE в RAPID WATCH/ от външен клиент /Delphi/ до сървъра, написан на ABB RAPID
- вид на данните – низ /String/ ,числа от 0 до 9 и букви А...Я и А…Z



Фиг.6.30. Предаване на данни

6.3. Изводи

Разработената в **Глава 4** методика, е използвана за програмиране на ПР в средата на Fanuc ROBOGUIDE и ABB RobotStudio. Анализът на получените при това резултати, доказва нейната функционалност.

С използване на Fanuc ROBOGUIDE е разработен сървър, написан на KAREL, които може да се използва, с клиентска програма, написана на почти всеки език за програмиране от високо ниво като: C, C++, C#, Java, Python, PERL, Delphi и др., спазвайки Socket Programming синтаксиса.

Чрез използване на езика от високо ниво Python, е създаден и тестван клиент, който да работи със създадения в средата на Fanuc ROBOGUIDE сървър.

С използване на ABB RobotStudio е генериран изходен код на RAPID, за програмиране на ПР изпълняващ “Pick and Place” операции за обслужване на конвейер.

Създаден е сървър написан на RAPID ABB, използван за трансфер на данни, при трансфер на данни и експлоатация на ПР.

ГЛАВА 7 ПРОГРАМИРАНЕ НА ПР ПРИ НАЛИЧИЕ НА ДОПЪЛНИТЕЛНА ОС

7.1. Програмиране на ПР при наличие на допълнителна ос

Използването на допълнителна ос на движение, осигуряваща мобилност на ПР, с цел разширяване на неговата работна зона, е често използвана практика при проектиране на роботизирани клетки. Управлението на тази допълнителна ос, трябва да се свърже към контролера на ПР, за да може тя да се управлява от програмата, по която работи ПР.

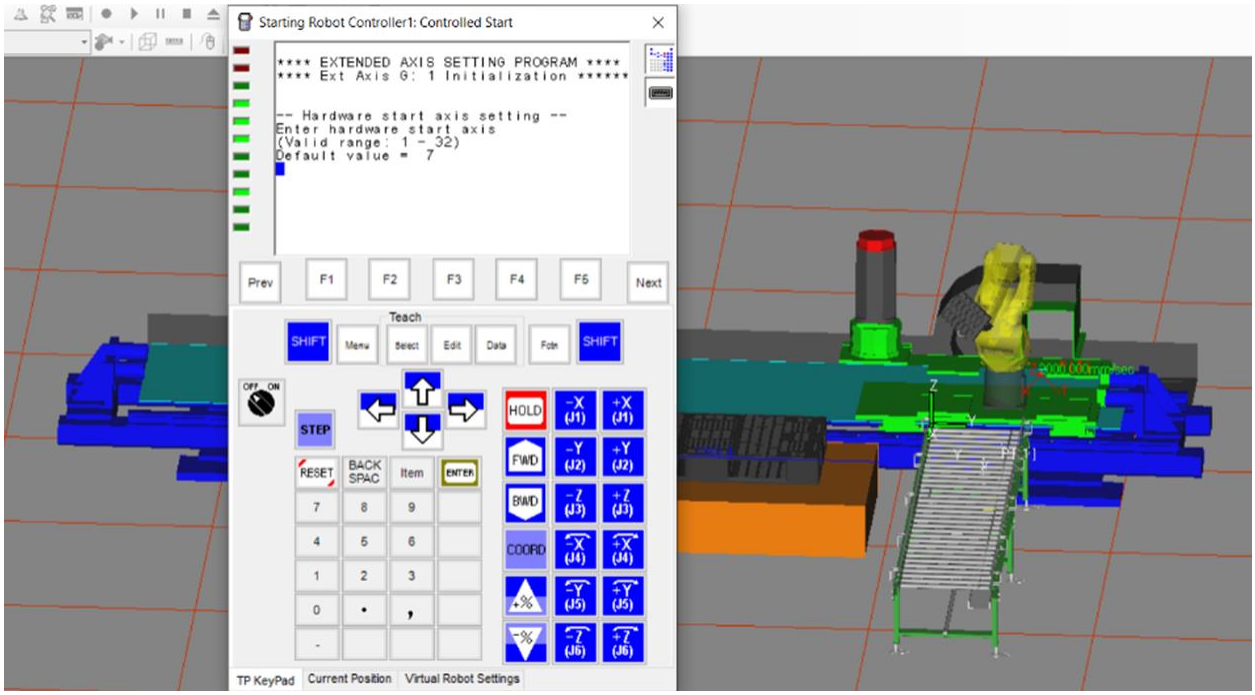
Наличието на допълнителна ос на движение, свързана към контролера на ПР, изисква допълнителни стъпки при разработване на УП за ПР с използване на CARC системи.

7.2. Програмиране на ПР при наличие на допълнителна ос в средата на ROBOGUIDE

ROBOGUIDE предоставя на потребителите различни инструменти за разработване на УП при наличие на допълнителна ос. За конфигуриране на допълнителната ос в средата на ROBOGUIDE, ще бъде използван виртуалния подвижен пулт на Fanuc LR Mate 200iD/7L, притежаващ пълната функционалност на реалния.

С използване на виртуалния подвижен пулт стартираме диалог за добавяне на допълнителна ос Фиг.7.3, като избираме

Maintenance - > Extended Axis Control - > F4(Manual) - >



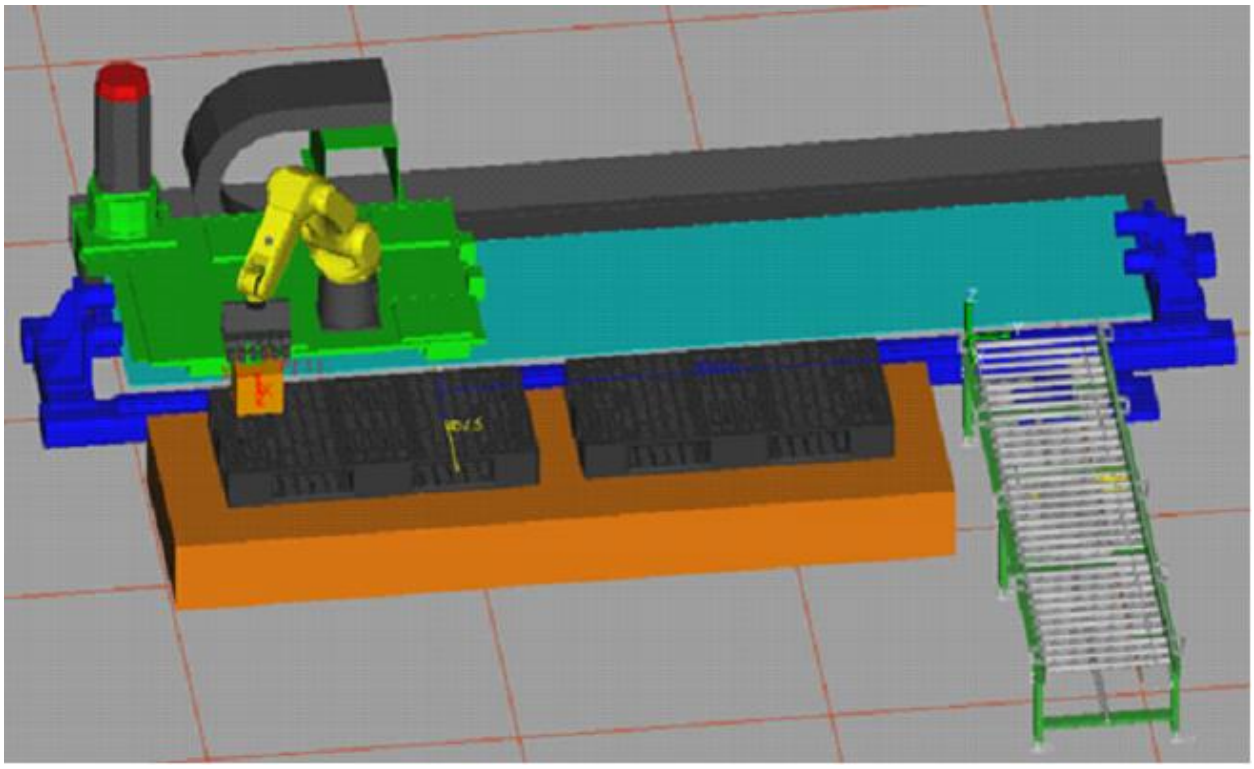
Фиг.7.3 Избиране на меню поддръжка

7.3. Програмиране на ПР при наличие на допълнителна ос в средата на RobotStudio

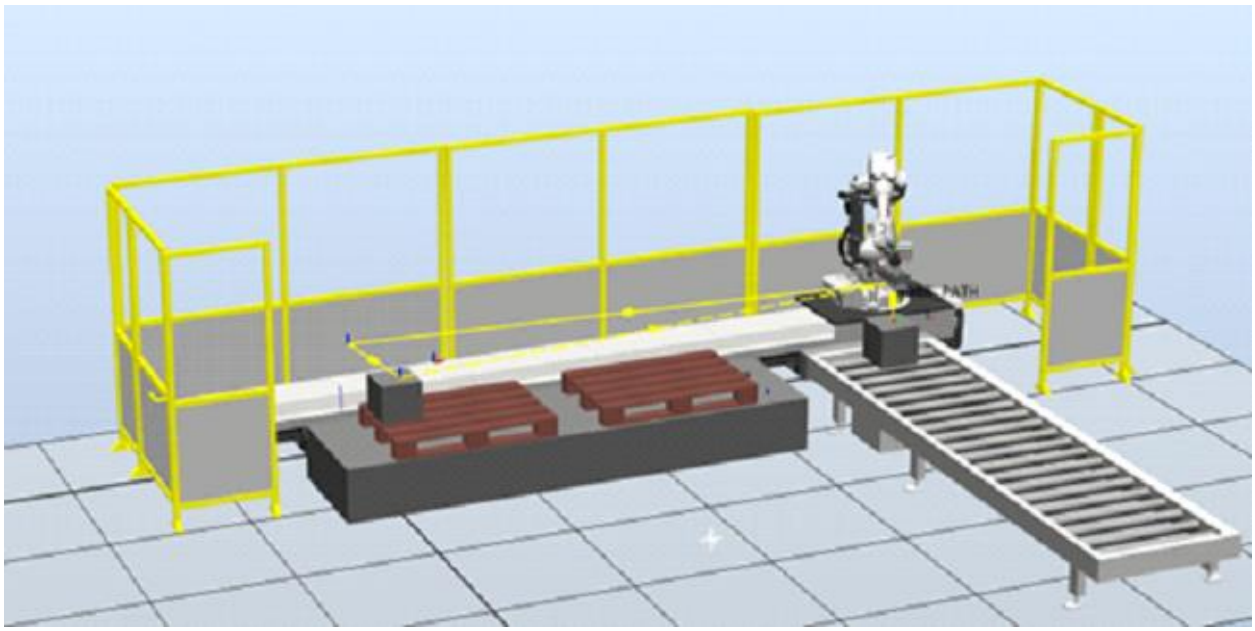
За конфигуриране на допълнителната ос в средата на RobotStudio, ще

бъдат използвани библиотеките, с индустриално оборудване, доставяни със системата.

3D модели на роботизирани клетки за изпълнение на “Pick and Place” операции с използване на допълнителна ос са показани както следва:



Фиг.7.16 Допълнителна ос в средата на FANUC ROBOGUIDE



Фиг.7.17 Допълнителна ос в средата на ABB RobotStudio

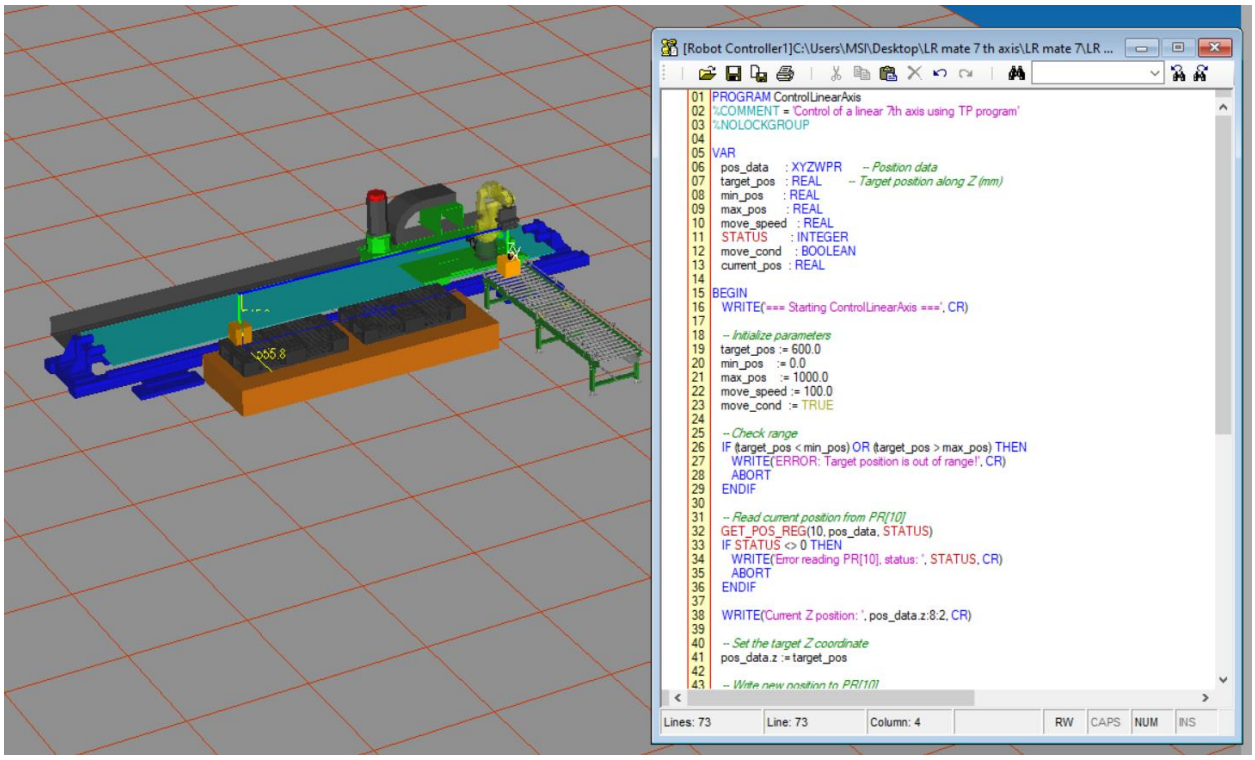
- ПР Fanuc LR Mate 200iD/7L и допълнителна ос в средата на Fanuc ROBOGUIDE - Фиг.7.16;
- ПР ABB IRB 2600 и долна подвижна релса IRBT-2005 в средата на ABB RobotStudio - Фиг.7.17.

7.4. Програмиране на ПР при наличие на допълнителна ос с използване на KAREL

В зависимост от конфигурацията на ПР, допълнителната ос може да бъде активирана в контролера, като се зададат съответните параметри. KAREL, като език за програмиране на ПР Fanuc, можете да контролирате всички оси чрез команди за движение и позициониране.

Fanuc KAREL по подразбиране не предоставя директни команди за движение, като LMOVE или JMOVE. Директното задаване на движение от KAREL е възможно само при активиран лиценз Motion Option (софтуерна опция J601).

По долу е дадена примерна програма на KAREL фиг.7.18, за управление на седма ос:



Фиг.7.18 KAREL програма за управление на седма ос в програмна среда Fanuc Roboguide

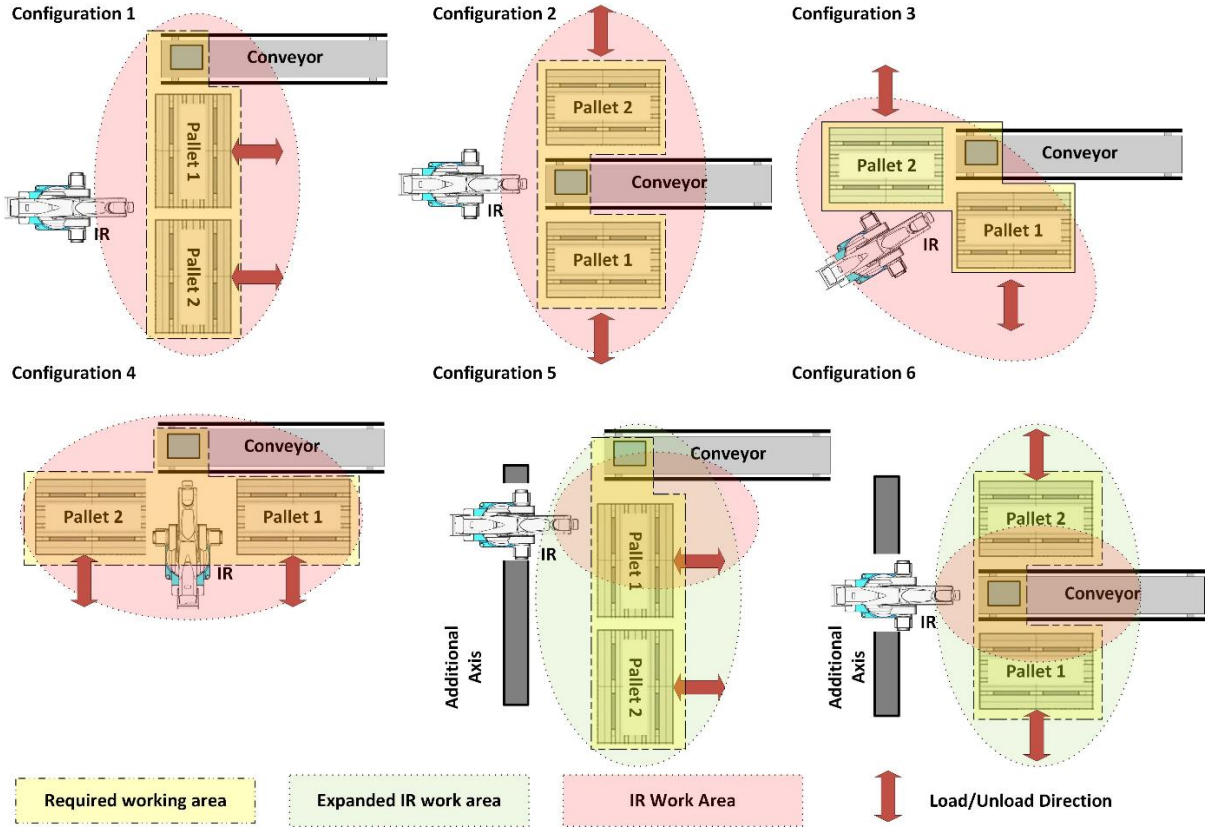
7.5. Проектиране на роботизирана клетка за изпълнение на “Pick and Place” операции в средата на ROBOGUIDE

С усложняването на изпълняваните от ПР операции и развитието на компютърните технологии, CARC системите, се превръщат в стандартен за инженерната практика инструмент. Тези системи непрекъснато се развиват, като освен разработване на програми за ПР, предоставят на техните потребители възможност за включване на ПР в структурата на различни роботизирани клетки.

За демонстриране на тези възможности ще бъде разгледано проектирането на роботизирана клетка за “Pick and Place” операции, състоящи се в палетизиране на кашони с размери 300x250x200 [mm] и тегло 5 [kg], в четири реда върху европалети EUR 2 1200x1000 [mm].

На Фиг.7.19 са показани шест възможни конфигурации на проектираната клетка, с отчитане на размерите на работната зона на използвания ПР. Клетката включва следните основни компоненти:

- конвейер - доставя кашоните до роботизираната клетка за палетизиране;
- промишлен робот - извършва подреждането на кашоните върху палатите;
- палети – определят зоната за подреждане на кашоните;
- допълнителна ос – осигурява разширяване на работната зона на ПР.

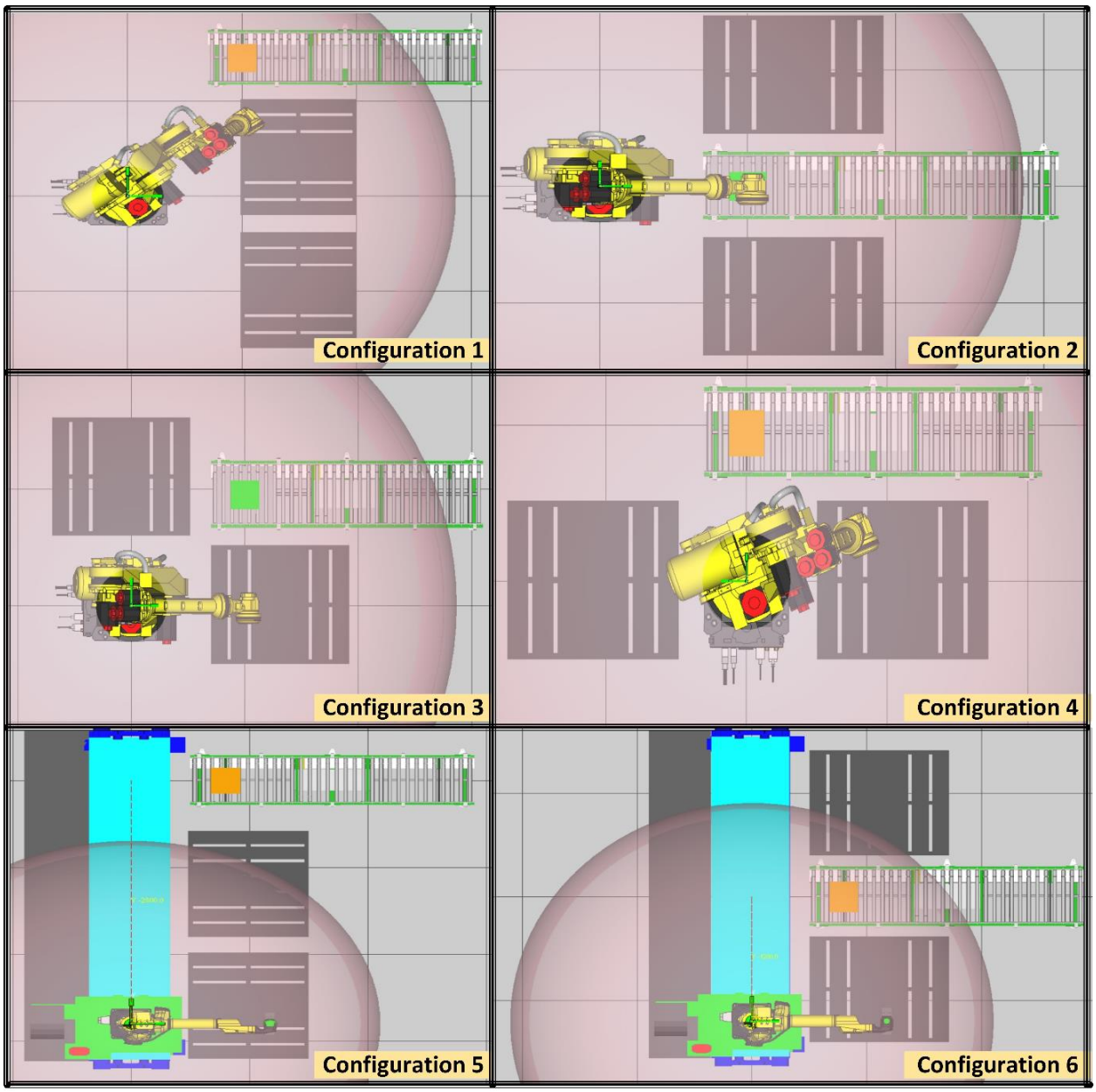


Фиг.7.19 Конфигурации на проектираната клетка

Анализът на разработените конфигурации, включва оценка на възможностите на избрания ПР да извършва необходимите операции при конкретното разположение на компонентите на клетката. При необходимост ПР може да бъде заменен с друг или да се добави допълнителна ос на мобилност, с цел разширяване на работната му зона.

Резултатът от изпълнение на тази стъпка за конфигурациите от Фиг.7.19 с използване на ROBOGUIDE, за проектираната роботизирана клетка е показан на Фиг.7.22.

Резултатът от анализа показва, че максималните размери на работните зони на избраните в стъпка **(3)** ПР, позволяват извършването на необходимите операции.



Фиг.7.22 Анализ на разработените конфигурации в средата на ROBOGUIDE

Табл.7.2 Характеристики на разработените конфигурации

Конфигурация	Време за подреждане на кашоните [min]			Обща площ [m^2]	Цена
	Палет1	Палет 2	Общо		
1	7.58	10.91	18.49	7	1
2	7.58	7.58	15.16	8	1
3	7.58	7.48	15.06	6	0,92
4	10.78	10.78	21.56	9	0,93
5	4.86	6.20	11.06	11	0,77
6	4.86	4.86	9.72	11	0,77

Данните в Табл.8.1 показват, че съгласно приетите критерии:

- с най-малко време за подреждане на кашоните върху двата палета (най-висока производителност е **конфигурация 6** - 9.72 [min];
- с най-малко заемана площ е **конфигурация 1** 7 [m^2];
- с най-ниска цена са **конфигурации 5 и 6**.

Въз основа на тези данни оптимална е **конфигурация 6**, която е избрана за използване в следващите етапи на проектиране.

7.6. Изводи

Използването на допълнителна ос на движение, осигурява мобилност на ПР, с цел разширяване на неговата работна зона.

Дефинирани са допълнителни стъпки, при разработване на УП за ПР с използване на CARC системи, при наличие на допълнителна ос на движение.

С използване на виртуален подвижен пулт, на Fanuc LR Mate 200iD/7L в средата на ROBOGUIDE, е направено конфигуриране на допълнителна ос на движение.

Направено е конфигуриране на допълнителната ос на движение на ПР в средата на RobotStudio, чрез използване на библиотеките, с индустриално оборудване, доставяни със системата.

Дефинирани са основните стъпки при проектиране на роботизирани клетки с използване на CARC системи.

Тези стъпки са приложени при проектиране на роботизирана клетка за изпълнение на “Pick and Place” операции в средата на ROBOGUIDE. В резултат на това са разработени шест конфигурации на проектираната клетка.

С помощта на ROBOGUIDE са определени характеристики, използвани за оценка на конфигурациите. Въз основа на направената оценка е избран оптимален вариант на проектираната клетка, съгласно критериите на заданието.

Получените резултати, показват, че съвременните CARC системи са мощен инструмент не само програмиране на ПР, но и за проектиране на роботизирани клетки.

Дефинираните стъпки са универсални и могат да се използват при работа с различни CARC системи.

НАУЧНО-ПРИЛОЖНИ И ПРИЛОЖНИ ПРИНОСИ

НАУЧНО ПРИЛОЖНИ ПРИНОСИ

НПП.1 Разработена е универсална методика, за генериране на управляващи програми за промишлени роботи с използване на CARC системи и са анализирани възможностите за използване на API при нейното прилагане.

НПП.2 Разработена е класификация по три признака на видовете атаки срещу роботизирани производствени системи и са дадени препоръки за защита от тези атаки.

ПРИЛОЖНИ ПРИНОСИ

ПП.1 Направен е сравнителен анализ на най-често използваните езици за програмиране от високо ниво, използвани в API при експлоатация на ПР и областите на тяхното приложение.

ПП.2 Разработени и сравнени са сървърен и клиентски код за TCP/IP свързване с използване на езиците за програмиране на ПР FANUC KAREL и ABB RAPID.

ПП.3 Разработен и анализиран е код, за криптиране на данни при експлоатация на ПР, с използване на различни езици за програмиране от високо ниво, поддържани в API при експлоатация на ПР.

ПП.4 Разработен е сървър, написан на KAREL, които може да се използва, с клиентска програма, написана на различни езици за програмиране от високо ниво, поддържани в API при експлоатация на ПР.

ПП.5 Дефинирани са основните стъпки при проектиране на роботизирани клетки с използване на CARC системи, които са универсални и могат да се използват при работа с различни CARC системи**.**

СПИСЪК НА ПУБЛИКАЦИИТЕ ПО ДИСЕРТАЦИОННИЯ ТРУД

1. Владимиров Б., *Мрежова сигурност при отдалечено програмиране на промишлени роботи*, Автоматизация на дискретното производство, бр. 3 юли 2021, стр. 171-175, ISSN 2682-9584

2. Владимиров Б., Ст. Николов, Сл. Димитров, *Приложение на промишлени роботи в опасни за човешкото здраве производствени среди*, Автоматизация на дискретното производство, бр. 4 юли 2022, стр. 94-98, ISSN 2682-9584

3. Vladimirov B., Nikolov St., Tsolov S., *Programming Industrial Robots in the Fanuc ROBOGUIDE Environment*, Engineering Proceedings, DOI: 10.3390/engproc2024070020, 2024 - Q4, Scopus

4. Vladimirov B., Nikolov St., *Remote transmission of information during the operation of robots Fanuc by using Fanuc ROBOGUIDE*, Annals of DAAAM and Proceedings of the International DAAAM Symposium, pp. 332-339, https://doi.org/10.2507/35th.daaam.proceedings.045, 2024 – Scopus

5. Nikolov St., Vladimirov B., *Designing robotic cells in the environment ROBOGUIDE*, Proceeding of 34th International Scientific and Technical Conference Automation of Discrete Production Engineering 2025, Az-buki National Publishing House, Bulgaria, https://doi.org/10.53656/adpe-2025.06

ЦИТИРАНЕ НА ПУБЛИКАЦИИТЕ ПО ДИСЕРТАЦИЯТА

Публикация 3 е цитирана в:

1. Taj, S., Awasthi, S., Dahri, H., Hashsham, S., Khan, R., *The Evolution of Industrial Automation and Cybersecurity Risks*, Advancing Cybersecurity in Smart Factories through Autonomous Robotic Defenses Book Chapter, 2025, ISBN 979-833730585-1, 979-833730583-7, Pages 397 – 429, DOI: 10.4018/979-8-3373-0583-7.ch015

Публикация 4 е цитирана в:

1. Sokolov, O., V., Andrusyshyn, A., Iakovets, V., Ivanov, *Intelligent Human–Robot Interaction Assistant for Collaborative Robots*, Electronics (Switzerland) Article Open Access, 2025 Multidisciplinary Digital Publishing Institute (MDPI), ISSN 20799292, Volume 14, Issue 6, Article number 1160, DOI: 10.3390/electronics14061160

2. Di, Gai, Xu, Weiyan, *Simulation Design of Industrial Robot Handling Workstation Based on ROBOGUIDE*, Lecture Notes in Electrical Engineering, Conference Volume 1441 LNEE, Pages 242 – 249, Book Series ISSN 18761100, Springer Science and Business Media Deutschland GmbH, 2025, ISBN 978-981968002-3, DOI: 10.1007/978-981-96-8003-0_27

Научноизследователски проект в помощ на докторант, към НИС на ТУ- София 2024 г., договор №241ПД0024-06
Тема: „Изследване на възможностите за програмиране на промишлени роботи с използване на API“
Ръководител: проф. д-р инж. Стилиян Николов Николов
Докторант: маг. инж. Борян Чавдаров Владимиров

SUMMARY

PROGRAMMING OF INDUSTRIAL ROBOTS USING API

Eng. Boryan Chavdarov Vladimirov MSc

In the dissertation thesis, the possibilities provided by different computer systems for using API in the operation of IR (Industrial Robots) are investigated.

In the first chapter, a brief historical overview of the history of IR, control systems, and programming methods used in the operation of IR is made. The main functions of the application programming interface API (Application Programming Interface) are reviewed.

In the second chapter, high-level programming languages used in automated programming of IR are reviewed. A comparative analysis of the capabilities of the programming languages ST, ABB, RAPID and Fanuc KAREL is performed. The use of programming languages Fanuc Karel and ABB RAPID in automated programming of IR is analysed.

The third chapter examines the main communication protocols through which remote operation of IR can be achieved. Server and client code for TCP/IP connection are developed and compared using IR programming languages Fanuc KAREL and ABB RAPID.

A classification of the types of attacks against robotic production systems is developed, and recommendations for protection against these attacks are provided.

In the fourth chapter, a comparative analysis of the functional capabilities of various modern CARC systems is performed, based on ten criteria. A universal methodology, including eight stages, is developed for generating Control Programs (CPs) for IR using CARC systems.

The fifth chapter discusses the use of API in the operation of IR. Some of the main characteristics when using languages Fanuc KAREL, ABB RAPID, C++, C#, Python and PERL in the exploitation of IR are compared. Code for data encryption in the operation of IR, using various high-level programming languages used in the API during IR operation, is developed and analysed.

In Chapter 6, the methodology developed in Chapter 4 is used for programming IR in the environment of Fanuc ROBOGUIDE and ABB ROBOT STUDIO, and the obtained results are analysed.

In Chapter 7, the specifics of programming IR with an additional axis are discussed. The main steps in designing robotic cells using CARC systems are defined. These steps are applied to the design of a robotic cell for performing “Pick and Place” operations in the environment of Fanuc ROBOGUIDE.

TECHNICAL UNIVERSITY – SOFIA

Faculty of Mechanical Engineering

Department of Automation of Discrete Manufacturing

Eng. Boryan Chavdarov Vladimirov MSc

PROGRAMMING INDUSTRIAL ROBOTS USING API

AUTHOR ABSTRACT

of a dissertation for the acquisition of an educational and scientific degree

"DOCTOR"

Field: 5. Technical Sciences

Professional field: 5.1 Mechanical Engineering

Scientific specialty: Production Automation

Scientific supervisor: Prof. Stiliyan Nikolov PhD

SOFIA, 2025

The dissertation was discussed and approved for defense by the Department Council of the Department of Automation of Discrete Production Engineering at the Faculty of Mechanical Engineering of TU-Sofia at a regular meeting held on November 10, 2025.

The public defense of the dissertation will take place on January 29, 2025, at 1:00 p.m. in the Conference Hall of the Library and Information Center of the Technical University of Sofia at an open meeting of the scientific jury, appointed by Order No. OZ-5. 1-104 / 04.12.2025 of the Rector of TU-Sofia, composed of:

1. Prof. Pancho Tomov, PhD – Chair
2. Assoc. Prof. Velizar Zaharinov, PhD – Scientific Secretary
3. Prof. Mladen Milushev, PhD
4. Prof. Nikolay Stoimenov, PhD
5. Assoc. Prof. Ivanka Peeva, PhD

Reviewers:

1. Assoc. Prof. Velizar Zaharinov, PhD
2. Assoc. Prof. Ivanka Peeva, PhD

The defense materials are available to interested parties at the office of the Faculty of Mechanical Engineering at TU-Sofia, block No. 4, room No. 3242.

The doctoral student is a part-time doctoral student at the Department of Automation of Discrete Production Engineering at the Faculty of Mechanical Engineering. The research for the dissertation was conducted by the author, some of which was supported by research projects.

Author: Eng. Boryan Vladimirov, MSc

Title: Programming Industrial Robots Using API

Circulation: 30 copies

Printed at the Technical University of Sofia

I. GENERAL CHARACTERISTICS OF THE DISSERTATION

Relevance of the problem

Modern industrial robots (IR) are highly energy efficient and capable of performing a wide range of tasks. This makes their use in various areas of industry increasingly attractive and cost-effective, ensuring the automation of various production activities and a safe working environment.

The development of computer technologies has made it possible, through the use of Application Programming Interface (API), to create user applications that automate the development of control programs and their transfer to the IRs used.

All this, along with the need for highly qualified personnel for programming IR, makes the issues related to researching the possibilities for automated and remote programming of IR relevant.

Purpose of the dissertation, main tasks, and research methods

The aim of this dissertation is to investigate the possibilities offered by various computer systems in the development of control programs for industrial robots, their transfer to industrial robots, and the possibilities for using APIs in the operation of these systems. To achieve this goal, the following tasks are to be accomplished:

1. Analysis of methods for programming industrial robots - IR
2. Analysis of the capabilities of different computer systems in the development of control programs for programming industrial robots
3. Development of a methodology for developing control programs for industrial robots using computer systems
4. Research on the capabilities that API provides for programming industrial robot using computer systems
5. Analysis of methods and means for data transfer in remote programming of IR
6. Development of an API application and data transfer system for remote programming of PR
7. Application of the developed applications and evaluation of their functionality

The research was conducted in a digital environment using computer and communication technologies

Scientific novelty

- A universal methodology has been developed for generating control programs for IR using CARC systems.
- The possibilities for using API in programming and operating IR have been determined.
- A classification of hacker attacks against robotic production systems and the problems they cause has been developed, and recommendations for protection against these attacks have been provided.

Practical applicability

Using the API, the following have been developed: server and client code for TCP/IP connection; a server written in KAREL and code for data encryption when operating the IR/Industrial robot/. The basic steps in designing robotic cells using CARC systems have been defined.

The results obtained are used in the training of students in the Department of Discrete Manufacturing Automation.

Approval

The results were validated using the IR/Industrial robot/ and programming software available at the Department of Discrete Manufacturing Automation.

Publications

The main achievements and results of the dissertation are published in five scientific publications, one of which is independent, while the others are co-authored with the scientific supervisor and colleagues from the Department of Automation of Discrete Manufacturing.

Three of the publications were presented in Bulgaria at the International Scientific Technical Conference "Automation of Discrete Manufacturing," and two at scientific forums abroad.

Two of the publications are indexed in Scopus.

Structure and capacity of the dissertation

The dissertation is **193** pages long and includes an introduction, seven chapters addressing the main tasks, a list of main contributions, a list of publications related to the dissertation, and a bibliography. A total of **138** literature sources are cited, **58** of which are in Latin and **7** in Cyrillic, while the rest are Internet addresses. The work includes a total of **89** figures and **13** tables. The numbers of the figures and tables in the abstract correspond to those in the dissertation..

II. CONTENTS OF THE DISSERTATION

CHAPTER 1 OVERVIEW AND ANALYSIS

1.1. History of the development of industrial robots

The use of robots in industry became of paramount importance in the last century. The stages of industrial robot evolution are examined.

1.2. Types of Industrial Robots(IR) and Their Areas of Application

Although industrial robots vary in design depending on the tasks they perform, the most common are the so-called automated „arm's". These robots can be classified into several different categories based on movement, application, architecture, model, etc.
The main types of industrial robots are discussed according to the International Federation of Robotics [33].

1.3. Industrial Robot Control Systems

The building blocks of control systems used in work and robotic devices operating in both industrial and non-industrial environments are examined, according to ISO 8373:2012 (revised in ISO 8373:2021 [38]).

1.4. Programming of the Industrial robot

Programming involves defining the desired actions and movements of the industrial robot so that it can perform them without operator intervention. The interpretation of the control program in the robot is performed by the controller. It is the part of the robot that connects the operator/programmer with the other elements (external and internal sensors, peripheral equipment, etc.). The main approaches used in robot programming are discussed.

1.5. Automating the process of generating control programs using API

An application programming interface (API) is a tool that enables integration between different applications by allowing users to use a programming language within another application.

Such applications can automate certain common or repetitive tasks that users perform. Other applications can combine commands in such a way as to provide additional functionality tailored specifically to the needs of certain companies and industries.

Computer-aided engineering (CAE) systems offer users various options for using APIs, which can be divided into two groups.

1.6. Conclusions

Industrial robots play a key role in the automation of industrial processes. They perform a variety of operations: welding, material handling, assembly, painting, and many others, with the complexity of the tasks performed constantly increasing.

The use of industrial robots is transforming most industrial systems. This transformation is influenced by:

- Implementation of the Internet of Things - more and more, robots will use intelligent sensors to collect information and improve process efficiency.
- Use of big data analysis - the data collected by industrial robots must be organized into systems that facilitate their analysis and reporting on their status.
- Use of virtual solutions - virtual solutions make it possible to analyze robotic systems without interrupting production.
- Use of open automation architectures - coordination between manufacturers is necessary to create standards and documentation that facilitate the introduction of industrial robots into a real production environment.

The ability to easily and effectively program industrial robots is critical to their use and the optimization of production processes.

The use of different computer systems and software platforms that enable the programming, simulation, and control of industrial robots significantly speeds up the process of controlling industrial robots and reduces the likelihood of errors.

Issues related to the use of different computer systems in the control of industrial robots are becoming increasingly relevant in connection with the concept of Industry 4.0.

Based on the review and analysis, the objective of this dissertation is defined as investigating the possibilities offered by different computer systems for using APIs in the operation of industrial robots.

CHAPTER 2 HIGH-LEVEL PROGRAMMING LANGUAGES USED IN AUTOMATED PROGRAMMING OF INDUSTRIAL ROBOT

2.1. FANUC KAREL programming language

Fanuc KAREL [43] is a programming language developed in the 1980s by GMF (General Motors Fanuc) Robotics, a joint venture between the Japanese company Fanuc and the American automotive giant GM (General Motors). GMF Robotics was one of the first companies to begin mass production of industrial robot for the automotive industry.

2.2. Structure of the KAREL language

KAREL includes structures and models common to high-level languages, as well as functions developed specifically for Fanuc PLC control. It features strictly typed variables, constants, custom types, procedures, functions, and provides access to various built-in functions that may not be available through the TP (Teach Pendant).

2.3. ABB RAPID programming language

The RAPID language [44] was created by ABB Robotics in the early 1990s as part of the development of a new generation of PRs. It was designed to make PR programming easier, more intuitive, and more powerful by providing better control over movements, inputs/outputs, and error handling. RAPID was introduced alongside the S4 controller and has undergone improvements and enhancements over the following decades, particularly with the IRC5 controllers, which added features such as network communication, IoT integration, and AI capabilities.

Table 2.3 Characteristics of FANUC KAREL and ABB RAPID

Characteristics	**Fanuc KAREL**	**ABB RAPID**
Platform	Industrial robot Fanuc	Industrial robot ABB
Language type	Procedural, highly typed, similar to Pascal/Ada	Procedural, simplified
Program format	PROGRAM name ... END name;	MODULE name ... ENDMODULE
Declaring variables	VAR speed : INTEGER;	VAR num speed;
Data type	INTEGER, REAL, BOOLEAN, STRING[n], ARRAY, XYZWPR, TRANS	num, bool, string, array, robtarget, jointtarget, pose
Conditional operators	IF ... THEN ... ELSE ... ENDIF;	IF ... THEN ... ELSE ... ENDIF
Loops	FOR ... ENDFOR; WHILE ... ENDWHILE;	FOR ... ENDFOR; WHILE ... ENDWHILE;
Functions and procedures	FUNCTION name : type ... END name; ROUTINE name ... END name;	FUNC type name ... ENDFUNC; PROC name ... ENDPROC;
Robot movement	Via TP programs or position registers GET_POS_REG, SET_POS_REG	MoveJ p1, v100, z10, tool1; MoveL p2, v50, fine, tool1;
Input/Output (I/O)	Access through built-in system procedures (IO_STATUS, SET_PORT_VAL)	SetDO do1, 1; WaitDI di1, 1;
Error handling	BEGIN ... ERROR ... END	TRAP ... ENDTRAP ERROR ... ENDERROR

Table 2.4 Comparison of the programming languages ST, ABB RAPID, and Fanuc KAREL

Characteristics	ST (Structured Text)	ABB RAPID	FANUC KAREL
Platform	PLC (Siemens, Beckhoff, Allen-Bradley, Schneider, etc.)	ABB industrial robots (IRC5 controllers)	FANUC industrial robots (R-J3, R-30iA, R-30iB controllers)
Language type	High level, standardized (IEC 61131-3), similar to Pascal/Modula/C	Specialized language for industrial robot, procedural	Specialized language for PR, procedural, similar to Ada/Pascal
Syntax	Similar to Pascal and C; standard for PLC	Simplified, intuitive for robotic movements	Highly structured, similar to Ada/Pascal
Ease of learning	Medium – requires PLC knowledge	High – easy for operators and programmers	More complex – requires more in-depth knowledge
Flexibility	Versatile – process control, logic, communications	Medium – powerful for robots, limited outside this area	Low – works only with FANUC controllers
Usage	PLC programming: logical operations, machine control, network communication	ABB IR programming: movements, logic, sensors, integration	FANUC IR programming: logic, sensors, advanced functions, system control
Robot movements	Through external libraries, limited support	Built-in and powerful motion control: MoveJ, MoveL, MoveC	Indirectly via position registers (GET_POS_REG, SET_POS_REG) or TP programs
Program structure	Functions, procedures, loops	Modules, procedures (PROC/FUNC), objects	Programs, subroutines, ROUTINE/FUNCTION, global and local variables
Applications	Machine automation, processes, SCADA integration	Robotic applications: welding, assembly, palletizing, CNC, vision processing, networks	Advanced industrial robot programming: logic, sensors, vision, networks
Error handling	TRY...CATCH, standard exception mechanism	ERROR and TRAP blocks for processing	BEGIN ... ERROR ... END and ON ERROR constructs
Network support	Modbus, EtherNet/IP, PROFINET, and other standard PLC protocols	TCP, UDP, FTP, Socket communication	TCP, FTP (no UDP); limited network integration

2.4. Structure of the RAPID language

The RAPID programming language is similar to most ST programming languages and closely resembles the C language. It is structured and easy to learn, providing a rich set of instructions and functions for controlling the movements of the industrial robot, processing signals, communicating with external devices, and much more.

2.5. Comparative analysis of the capabilities of high-level programming languages used in automated programming of industrial robot

Fanuc KAREL and ABB RAPID are two different programming languages used to control PLCs. Although both languages have similar objectives, they differ in structure, syntax, and purpose. A comparison of the characteristics of the two languages is given in Table 2.3.

Despite the specifics of KAREL and RAPID, anyone familiar with C-style programming languages should find it relatively easy to program PLCs using these languages.

Table 2.4 provides a comparison of the ST, ABB RAPID, and Fanuc KAREL programming languages used in industrial automation.

2.6. FANUC KAREL and ABB RAPID in the context of Industry 4.0

The Fanuc KAREL and ABB RAPID programming languages provide additional capabilities for PR management, compatible with modern Industry 4.0 concepts such as the Internet of Things (IoT), Cloud Technologies (CT), Artificial Intelligence (AI), and intelligent automation.

2.7. Conclusions

The programming languages ST, ABB RAPID, and Fanuc KAREL, used in industrial automation, have some common features, but each of them is developed for a specific platform and has its own unique characteristics.

The additional capabilities that the KAREL and RAPID programming languages provide to users when programming PLCs are analyzed.

A comparative analysis of the capabilities of the ST, ABB RAPID, and Fanuc KAREL programming languages in automated PLC programming is made.

The use of the Fanuc KAREL and ABB RAPID programming languages is analyzed in the context of the use of: the Internet of Things; cloud technologies; artificial intelligence and intelligent automation in the operation of industrial robot.

CHAPTER 3 DATA TRANSFER DURING THE OPERATION OF THE INDUSTRIAL ROBOT

3.1. Data transfer protocols for the operation of industrial robots

Currently, there are many technical solutions used for remote programming and control of production equipment executing a given production program.

A comparison of the network functions and protocols supported in the high-level programming languages discussed in Chapter 2 used in the operation of industrial robot is given in Table 3.1.

Table 3.1 Network functions and protocols in ST, FANUC KAREL, and ABB RAPID

Function/ Protocol	ST (PLC)	FANUC KAREL	ABB RAPID	Notes/ limitations
Supported network protocols	TCP/IP, UDP, Modbus TCP, OPC-UA, MQTT, ProfiNet, EtherNet/IP	TCP/IP, FTP, HTTP, Socket Messaging	TCP/IP, UDP*, FTP, HTTP,	*RAPID: UDP is not built-in; requires external configuration; KAREL does not have UDP
Socket creation (TCP/IP client/server)	SysSockCreate(), SysSockBind(), SysSockConnect()	Communication via configured tags, MSG_DISCO and OPEN FILE	Web Services (REST/SOAP), Socket Messaging	UDP sockets on RAPID/KAREL require external configuration or scripts
Data sending	SysSockSend()	Socket Send Messaging	SocketCreate, SocketBind, SocketConnect	
Data receiving	SysSockRecv()	Socket Receive Messaging	SocketSend	
Socket closing	SysSockClose()	Closing via standard file operations	SocketReceive (included in the manual, but not always used)	
FTP communication	SysFTPClientPut(), SysFTPClientGet()	CALL_PROG	SocketClose	
HTTP/HTTPS support	SysWebClientRequest() (on some PLCs)	HTTP_REQ	Built-in FTP server, access via GetFTP	
UDP communication	SysSockSendTo(), SysSockRecvFrom()	Not supported	HTTP and Web Services (REST/SOAP)	KAREL does not have UDP; RAPID – only through additional configuration and scripts
WebSockets	Limited support	Limited support	UDPSend, UDPReceive	No universal built-in WebSocket support
Industrial protocols	OPC UA, Modbus TCP, ProfiNet, EtherNet/IP, MQTT	OPC UA (with license), Modbus TCP, EtherNet/IP, ProfiNet	Possible via Web Services	KAREL: OPC UA requires a license; RAPID – built-in, full support
JSON/XML support	Possible with libraries (depending on manufacturer)	Manual string processing	OPC UA (built-in), Modbus TCP, ProfiNet, EtherNet/IP	KAREL: manual processing; ST PLC – depends on available libraries
Built-in Web Server	Supported on some PLCs	Limited support	Built-in parsing for JSON/XML	KAREL: limited; RAPID – full, including ABB Ability interface
Cloud integration	MQTT, OPC-UA to cloud services	Requires external scripts – FANUC FIELD SYSTEM	Supports ABB Ability Cloud	KAREL: TP script or FIELD System required; ST PLC – depends on libraries

Table 3.2 Server code for TCP/IP connection

FANUC KAREL	ABB RAPID

PROGRAM tcp_server
%STACKSIZE = 4000
%NOLOCKGROUP
%NOPAUSE=ERROR+COMMAND+TPENABLE
%ENVIRONMENT uif
%ENVIRONMENT memo
%ENVIRONMENT flbt
%INCLUDE klevkeys
%ENVIRONMENT bynam
%INCLUDE klevkmsk
%ENVIRONMENT fdev
%INCLUDE klevccdf
%ENVIRONMENT kclop
%ENVIRONMENT sysdef
VAR
file_var : FILE
tmp_int : INTEGER
tmp_str1 : STRING[128]
tmp_int1 : INTEGER
tmp_str : STRING[128]
status : INTEGER
entry : INTEGER
BEGIN
SET_FILE_ATR(file_var, ATR_IA)
SET_VAR(entry, 192.168.1.3 '*SYSTEM*', '$HOSTS_CFG [3]. $SERVER_PORT', 6008, status)
WRITE TPPROMPT('Connecting.', status, CR)
MSG_CONNECT('S3', status)
WRITE TPPROMPT('Connect Status = ', status, CR)
IF status = 0 THEN
WRITE TPPROMPT('Opening', CR)
FOR tmp_int1 = 1 TO 20 DO
OPEN FILE file_var ('rw', 'S3:')
status = IO_STATUS(file_var)
WRITE TPPROMPT(status, CR)
IF status = 0 THEN
FOR tmp_int = 1 TO 1000 DO
WRITE TPPROMPT ('Reading', CR)
BYTES_AHEAD (file_var, entry, status)
WRITE TPPROMPT (entry, status, CR)
READ file_var (tmp_str::10)
status = IO_STATUS(file_var)
WRITE TPPROMPT(status, CR)
ENDFOR
CLOSE FILE file_var
ENDIF
ENDFOR
WRITE TPPROMPT('Disconnecting..', CR)
MSG_DISCO('S3:', status)
WRITE TPPROMPT('Done.', CR)
ENDI
END tcp_server
	MODULE Module1
	VAR socketdev serverSocket;
	VAR socketdev clientSocket;
	VAR string data;
	PROC main()
	!Add your code here
	SocketCreate serversocket;
	SocketBind serverSocket, “127.0.0.1” , 5000;
	SocketListen serverSocket;
	SocketAccept serverSocket, clientSocket, \Time:= WAIT_MAX;
	
	SocketReceive clientSocket \Str:=data;
	SocketSend clientSocket \Str:= “received”;
	SocketClose clientSocket;
	SocketClose serverSocket;
	ERROR
	IF ERRNO=ERR_SOCK_TIMEOUT THEN
	RETRY;
	IF ERRNO=ERR_SOCK_TIMEOUT THEN
	RETRY;
	ELSEIF ERRNO=ERR_SOCK_CLOSED THEN
	SocketClose clientSocket;
	SocketClose serverSocket;
	SocketCreate serverSocket;
	SocketBind serverSocket, “127.0.0.1” , 5000 ;
	SocketListen serverSocket;
	SocketAccpet serverSocket,clientSocket, \Time:=WAIT_MAX;
	RETRY;
	ELSE
	stop;
	ENDIF
	ENDPROC
	ENDMODULE

Table 3.2 provides a comparison of server code for TCP/IP connection developed with Fanuc KAREL and ABB RAPID.

Table 3.3 Client code for TCP/IP connection

FANUC KAREL	ABB RAPID

PROGRAM tcp_client
%STACKSIZE = 4000
VAR
file_var : FILE
status : INTEGER
recv_str : STRING[128]
send_str : STRING[16] := "Hello from client"
bytes_ahead : INTEGER
i : INTEGER
BEGIN
MSG_CONNECT('S3', status)
WRITE TPPROMPT('Connect Status = ', status, CR)
IF status = 0 THEN
OPEN FILE file_var ('rw', 'S3:')
status = IO_STATUS(file_var)
IF status = 0 THEN
FOR i = 1 TO 10 DO
WRITE TPPROMPT('Sending data...', CR)
WRITE file_var(send_str)
status = IO_STATUS(file_var)
IF status <> 0 THEN
WRITE TPPROMPT('Error sending data', CR)
EXIT FOR
ENDIF
 	
BYTES_AHEAD(file_var, bytes_ahead, status)
IF bytes_ahead > 0 THEN
READ file_var(recv_str::bytes_ahead)
WRITE TPPROMPT('Received: ', recv_str, CR)
ELSE
WRITE TPPROMPT('No data available to read', CR)
ENDIF
ENDFOR
CLOSE FILE file_var
ELSE
WRITE TPPROMPT('Error opening socket file', CR)
ENDIF
 	
MSG_DISCO('S3', status)
WRITE TPPROMPT('Disconnected', CR)
ELSE
WRITE TPPROMPT('Connection failed', CR)
ENDIF
END tcp_client
	MODULE Module1
	VAR socketdev clientSocket;
	VAR string data;
	VAR string recvData;
	VAR num errStatus;
	PROC main()
	! Create a client socket SocketCreate clientSocket;
	
	! Attempt to connect to the server at 127.0.0.1:5000
	SocketConnect clientSocket, "127.0.0.1", 5000 \Timeout:=5000; ! Check for connection errors
	IF ERRNO <> 0 THEN TPWrite "Connection failed with error: "; TPWrite ERRNO; RETURN; ENDIF "
	! Send message to server
	data := "Hello Server"; SocketSend clientSocket \Str:=data; ! Receive response from server
	SocketReceive clientSocket \Str:=recvData;
	! Displaying the received data TPWrite "Received from server: " & recvData;
	! Closing the client socket SocketClose clientSocket; ENDPROC
	ENDMODULE

When operating industrial robots, two interrelated groups of tasks can be distinguished: generating control programs and transferring data between the robot controller and the platform on which the remote programming and control software is installed.

3.2. Connecting industrial robot to a network

Connecting industrial robots to a network allows for their integration with existing automated production systems and also offers opportunities for remote control, monitoring, and data analysis.

Table 3.3 provides a comparison of client code for TCP/IP connection developed with Fanuc KAREL and ABB RAPID.

3.3. Network security during industrial robot operation

As industrial networks become more complex, threats to their security also increase, and designers of such technical solutions must take into account all types of threats in order to ensure the security of the transmitted information.

Remote programming of industrial robots is based on a secure network connection through information encryption and network devices that ensure the security of the transmitted information.

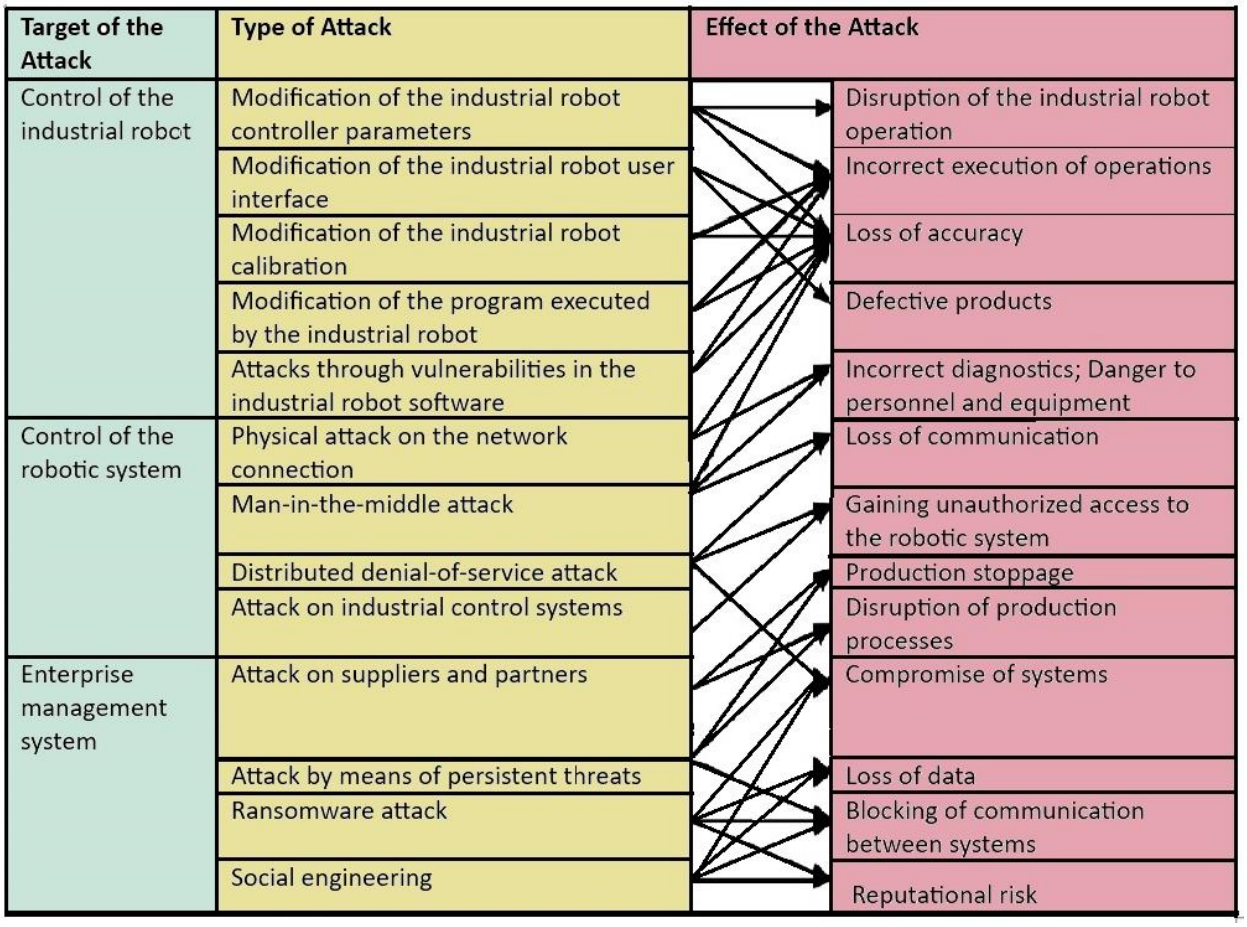


Fig. 3.2 Classification of attacks targeting robotic manufacturing systems

3.4. Attacks during industrial robot operation

Based on an analysis of the types of attacks against robotic production systems, a classification of these attacks has been developed according to three criteria, as shown in Fig. 3.2.

Table 3.4 Types of attacks against robotic systems

Type of attack	Defense approaches
Attacks against industrial robot control systems	
Change of parameters of the Industrial Robots controller	Access control, anomaly detection, parameter validation, two-factor authentication (2FA), reinstallation from backup
Change in the user interface of industrial robots	Access control, data monitoring, communication encryption, regular backups
Change in the calibration of Industrial Robots	Access control, calibration verification, anomaly detection, digital signatures, regular backups, recalibration
Change in the program executed by Industrial Robots	Control and restriction of access, backup copies of programs, program reloading
Attacks through vulnerabilities in Industrial Robot software	Access control, penetration testing to detect vulnerabilities, regular backups, regular software updates.
Attacks against the control of robotic manufacturing systems	
Physical attacks via the network connection	Control and restriction of access to network ports, protected areas, physical protection of the network, VPN protection
Man-in-the-Middle (MITM) attacks	Network traffic encryption, VPN additional firewall, daily backup of the robotic system, network segmentation
Distributed Denial-of-Service (DDoS) attacks	Network traffic monitoring, firewalls, DDoS protection, network segmentation
Attacks on industrial control systems	System monitoring and analysis, network segmentation, ICS system protection
Attacks against the company's management system	
Attacks against suppliers and partners - Supply Chain Attacks	Data reception network control, remote network monitoring, software verification, digital signatures
Attacks through persistent threats Advanced Persistent Threats	Monitoring network traffic to detect suspicious activity, training staff to identify persistent attacks, regular audits
Ransomware attacks	Monitoring for malicious code detection, regular data backups, antivirus protection, network segmentation
Social engineering	Security policies, security testing, restricting the sharing of sensitive information, verification of users requesting access

These signs are:

- **Attacks against the industrial robot administration**

The first group consists of attacks targeting the Industrial Robots controller. They aim to change the behavior of Industrial Robots during their operation.

- **Attacks against the control of robotic manufacturing systems**

The second group consists of attacks targeting the network to which the industrial robot controller is connected. Their aim is to penetrate the robotic system's control system and disrupt its operation.

- **Attacks against the company's management system**

The third group consists of attacks aimed at gaining access to robotic systems without the necessary rights, with the aim of spreading malware that disrupts the

operation of the systems.

The effects of the attack are: unauthorized access; production stoppage; data loss; financial losses; reputational risk.

Table 3.4 lists the approaches that can be used to protect against the attacks discussed.

3.5. Data encryption in the operation of industrial robots

The advantage of using open source systems is that they are free to be further developed and technologically advanced according to user needs.

When using encryption devices, the security of the transmitted information is ensured, thus preventing the possibility of loss or unauthorized access to sensitive information. At present, the security of industrial networks cannot be overlooked, as a result of which more and more affordable data encryption solutions are available on the market.

3.6. Conclusions

Connecting Industrial Robots (IR) to a network is key to modern industrial applications. The ability to communicate over a network not only improves productivity but also allows remote control and monitoring of the robotic system.

The main communication protocols through which remote operation of industrial robots can be achieved are examined, and the network functions embedded in the programming languages of Industrial Robots ST, Fanuc KAREL, and ABB RAPID are analyzed.

Server and client code for TCP/IP connection using the PR programming languages Fanuc KAREL and ABB RAPID are developed and compared.

A classification of the types of attacks against robotic production systems based on three criteria is developed, and recommendations for protection against these attacks are given.

Data encryption in PLC programming plays an important role in ensuring the security and confidentiality of communication between robotic systems and external devices. This is especially important when the robotic system is integrated into an industrial network where the exchange of sensitive data may be exposed to attacks and unauthorized access. Encryption helps protect data and communication channels, which is critical to maintaining the confidentiality and integrity of the system.

An example of socket communication protection during the operation of Fanuc Industrial Robots has been developed using the KAREL language.

CHAPTER 4 AUTOMATION PROGRAMMING OF INDUSTRIAL ROBOTS

4.1. Computer systems for programming industrial robots

These are CARC (Computer Aided Robot Control) [46] systems, which can also be found as OLPE (Off-line Programming Environments). They apply the latest

achievements in the field of computer technology and engineering computer graphics.

CARC systems use 3D models of IR and other production equipment, through which robotic production systems can be built, their operation can be simulated and analyzed, and control programs can be generated for the IR included in them.

4.2. Analysis of the capabilities of the main modern CARC systems

Taking into account the diversity of CARC systems currently offered by different manufacturers, a study of their functional capabilities was conducted using publicly available sources, based on the following ten criteria:

- availability of specialized programming modules for performing various PR tasks;
- availability of libraries with 3D models of various IR;
- availability of libraries with 3D models of end effector;
- availability of libraries with 3D models of production equipment;
- possibility of using a virtual control panel for IR management;
- possibility to use high-level industrial robot programming languages and API;
- possibility to simulate and optimize the operation of IR;
- possibility to design and analyze the operation of robotic cells;
- possibility to exchange data with other CAx systems;
- support for various IR controllers.

The results of the study are presented in Appendix 1. The data there show that the main differences between the CARC systems available on the market are related to their ability to exchange data with other CAx systems and support IR controllers offered by different companies.

4.3. Methodology for generating control programs for industrial robots using CARC systems

Regardless of the variety of CARC systems currently available, the main steps in working with these systems largely overlap. Here, a methodology is proposed that describes the main steps in working with CARC systems necessary for generating Control Programs (CP) for industrial robot, and the possibilities for using API in this process are analyzed.

The main steps required for generating CP for Industrial Robots (IR), regardless of the CARC system used, are given in the developed methodology shown in Fig. 4.1.

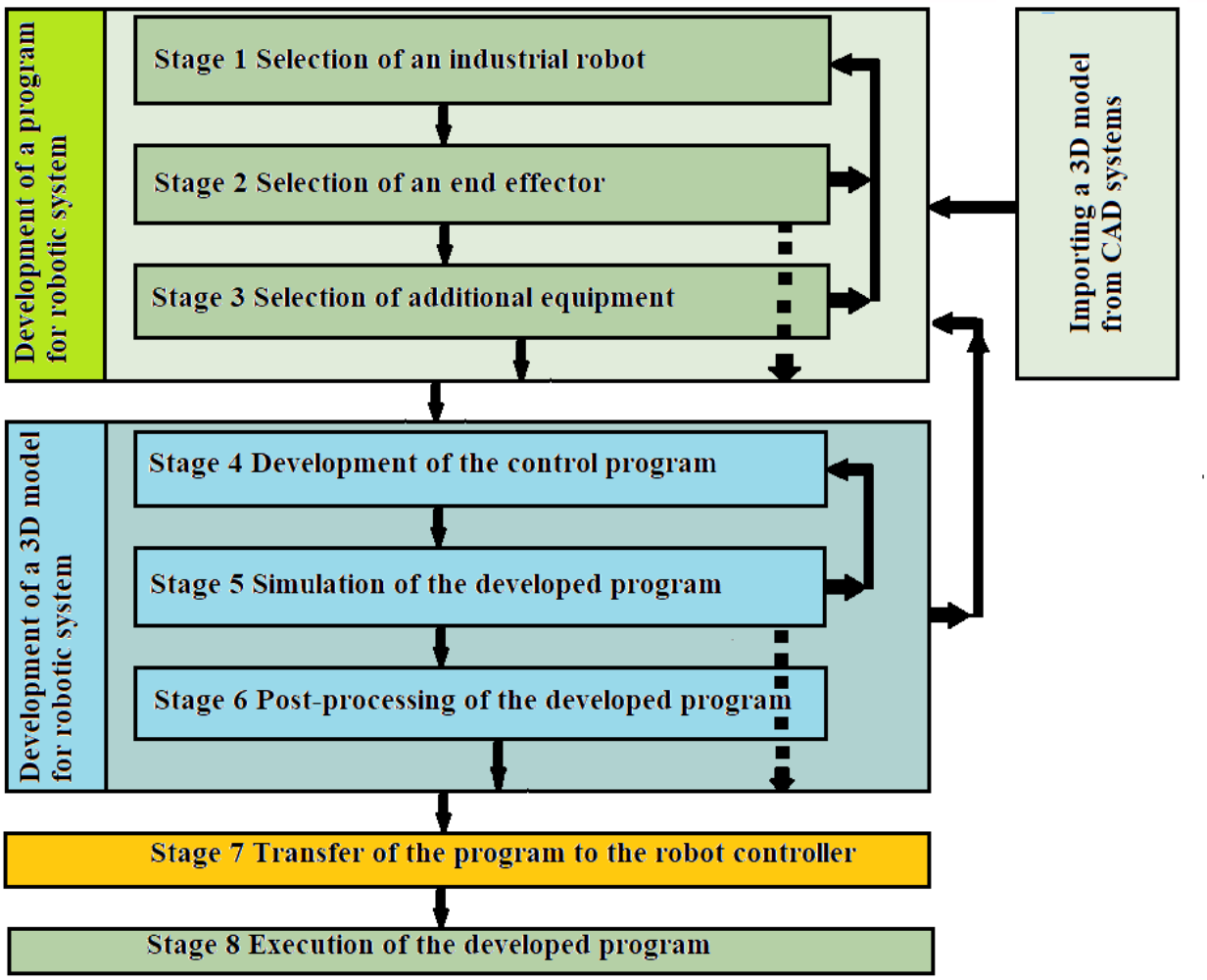


Fig. 4.1 Methodology for generating a IR management plan using CARC systems

They are as follows:

Stage 1 Selection of an industrial robot

Stage 2 Selection of an end effector

Stage 3 Selection of additional equipment

Stage 4 Development of the control program

Stage 5 Simulation of the developed program

Stage 6 Post-processing of the developed program

Stage 7 Transfer of the program to the robot controller

Stage 8 Execution of the developed program

4.4. Conclusions

A comparative analysis of the functional capabilities of various modern CARC systems available on the market has been performed based on ten criteria.

Modern CARC systems provide users with extensive capabilities for generating control program's for industrial robot's, allowing the use of various 3D models, virtual consoles, APIs, and high-level programming languages in the control system development process.

A methodology comprising eight stages has been developed for generating control program for industrial robot using CARC systems.

The main steps in using the developed methodology are discussed and the possibilities for using APIs in this process are analyzed.

The developed methodology for generating a control program's for IR using CARC systems is universal and does not depend on the CARC system used and the industrial robot for which the industrial robot is generated.

CHAPTER 5 USE OF API IN THE OPERATION OF INDUSTRIAL ROBOT

5.1. API in computer systems for programming industrial robot

Taking into account the individual steps in the proposed methodology and the analysis of the possibilities that modern CARC systems offer users for generating Control Programs/CP/ for Industrial Robots/IR/, the use of API in this process can be used for:

- developing software applications for creating 3D models of various components for designing robotic cells;
- developing applications for remote programming of IR;
- developing applications for remote data exchange between IR and a host computer.

The API provides an interface that allows different software applications to communicate and interact with each other. In the context of IR operation, the API plays a key role in the following aspects:

5.2. Programming languages used in the API when operating Industrial Robots - IR

Table 5.1 provides a comparison of some key characteristics when using Fanuc KAREL, ABB RAPID, C++, C#, Python, and Perl languages in the operation of industrial robots.

Table 5.1 Comparison of Fanuc KAREL, ABB RAPID, C++, C#, Python, and Perl languages in industrial robot operation

Function / Description	Fanuc KAREL	ABB RAPID	C/C++	C#	Python	Perl
Creating a socket	MSG_CONN / MSG_CONNECT	SocketCreate socketdev	socket()	TcpListener<	socket.socket()	IO::Socket::INET->new(...)
Connecting to a server	MSG_DISCO	SocketConnect socketdev, ip, port	connect()	TcpClient	socket.connect()	connect()
Sending data	KAREL commands	SocketSend socketdev, data	send()	NetworkStream.Write()	send()	print SOCKET "data"
Receiving data	WRITE	SocketReceive socketdev, buffer	recv()	NetworkStream.Read()	recv()	$data = <SOCKET>
Closing a socket	MSG_DISCO(tag_name, status)	SocketClose socketdev	close()	Close()	close()	close(SOCKET)
Supported protocols	TCP/IP	TCP/IP, UDP	TCP/IP, UDP	TCP/IP, UDP	TCP/IP, UDP	TCP/IP, UDP
Multi-client support	Limited server	Yes	Yes, through multithreading (pthread/std::thread)	Yes, through Task/Threading	Yes, through threading or asyncio	Yes, through fork()
Error handling	BEGIN ... ERROR	TRAP Handles errors through TRAP blocks	try-catch Handles errors through try-catch blocks.	try-catch Handles errors using try-catch blocks.	try-except Handles errors using try-except blocks.	eval and $@ Handles errors using eval and $@.
Applications in industrial automation	Processes errors via ERROR blocks Robot controllers, communication with PLC	SCADA, IoT, Web integrations	Embedded systems, communication with robots	Industrial applications, SCADA	IoT, AI, data processing, robots	Web services, automation
Function / Description	**FANUC KAREL**	**ABB RAPID**	**C/C++**	**C#**	**Python**	**Perl**

Table 5.2 provides a comparison of client code for TCP/IP connection developed with high-level programming languages C++, C#, Python, and Perl. Client code for TCP/IP when operating industrial robots, developed using the Fanuc KAREL and ABB RAPID languages, is given in Table 3.3.

Table 5.3 compares the use of different high-level programming languages used for data encryption in the operation of industrial robots.

5.3. Applications of API in industrial automation

The use of API in industrial automation can be considered in the following aspects:

- production automation - allows synchronization of the work of industrial robots and other production equipment;
- collaborative robots (Cobots) - used to control robots that work together with humans;

- smart factories - ensures the integration of industrial robots with IoT and machine learning systems.

Table 5.2 Comparison of client codes for TCP/IP connection

High-level language	TCP client – source code
Python	import socket client = socket.socket(socket.AF_INET, socket.SOCK_STREAM) client.connect(("192.168.1.100", 5000)) client.send(b"Hello, Server!") response = client.recv(1024) print("Received:", response.decode()) client.close()
PERL	use IO::Socket::INET; my $client = IO::Socket::INET->new(PeerHost => "192.168.1.100", PeerPort => "5000", Proto => "tcp") or die "Could not connect\n"; print $client "Hello, Server!\n"; my $response = <$client>; print "Received: $response\n"; close($client)

C++	#include <iostream>
	#include <sys/socket.h>
	#include <arpa/inet.h>
	#include <unistd.h>
	int main() {
	int sock = socket(AF_INET, SOCK_STREAM, 0);
	struct sockaddr_in server;
	server.sin_family = AF_INET;
	server.sin_port = htons(5000);
	inet_pton(AF_INET, "192.168.1.100", &server.sin_addr);
	connect(sock, (struct sockaddr*)&server, sizeof(server));
	send(sock, "Hello, Server!", 14, 0);
	char buffer[1024] = {0};
	recv(sock, buffer, 1024, 0);
	close(sock);
	return 0;
	}

C#	using System;
	using System.Net.Sockets;
	using System.Text;
	class Program {
	static void Main() {
	TcpClient client = new TcpClient("192.168.1.100", 5000);
	NetworkStream stream = client.GetStream();
	byte[] data = Encoding.ASCII.GetBytes("Hello, Server!");
	stream.Write(data, 0, data.Length);
	byte[] buffer = new byte[1024];
	stream.Read(buffer, 0, buffer.Length);
	Console.WriteLine("Received: " + Encoding.ASCII.GetString(buffer));
	client.Close();
	}
	}

5.4. Conclusions

The API in computer systems for programming industrial robots provides a powerful tool for integration, automation, and control. It facilitates the development of software solutions and ensures that industrial robots can be adapted to the dynamically changing requirements of modern industry.

The possibilities and aspects that API provides when generating UP for Industrial Robots and their operation are discussed.

The most commonly used high-level programming languages used in API when operating Industrial Robots are discussed and their areas of application are analyzed.

Code for data encryption in the operation of industrial robots has been developed and analyzed, using various high-level programming languages used in API in the operation of industrial robots.

A comparison has been made of some of the main characteristics of using the languages Fanuc KAREL, ABB RAPID, C++, C#, Python, and Perl in the operation of industrial robots.

A client-server system has been created using Common LISP and Emacs LISP.

CHAPTER 6 APPLICATION OF THE DEVELOPED METHODOLOGY FOR PROGRAMMING INDUSTRIAL ROBOTS

6.1. Programming Industrial Robots in the ROBOGUIDE Environment

The developed methodology was used for programming industrial robots in the ROBOGUIDE environment.

Using Fanuc ROBOGUIDE, the created TCP_SERVER.kl server written in KAREL was compiled (Fig. 6.9).

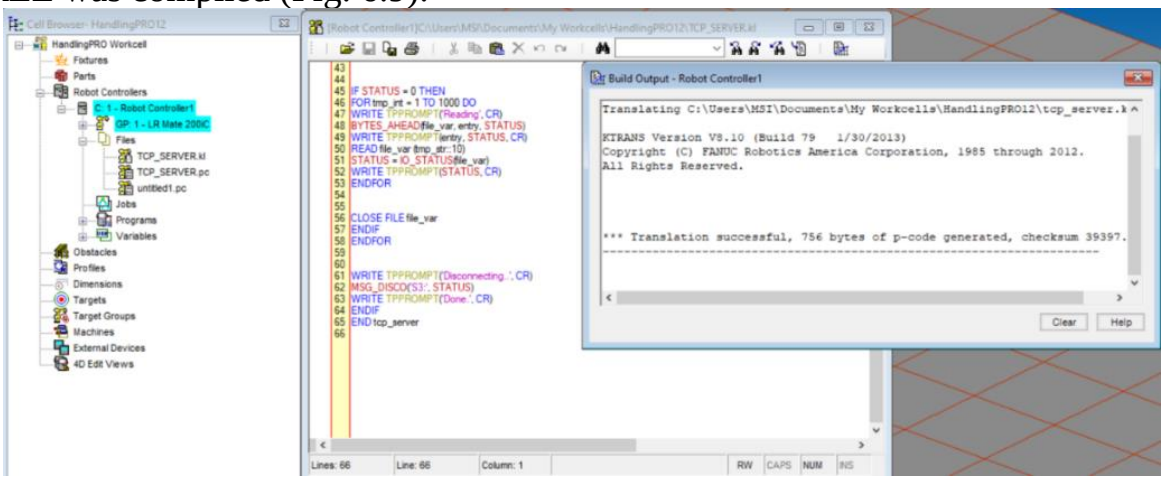


Fig. 6.9 Compiling TCP_SERVER.kl in Fanuc ROBOGUIDE

6.2. Programming Industrial Robots in the ABB RobotStudio Environment

The developed methodology was used for programming industrial robots in the ABB Robot Studio environment.

The manipulated item is a box with the desired dimensions and weight (Fig. 6.23).

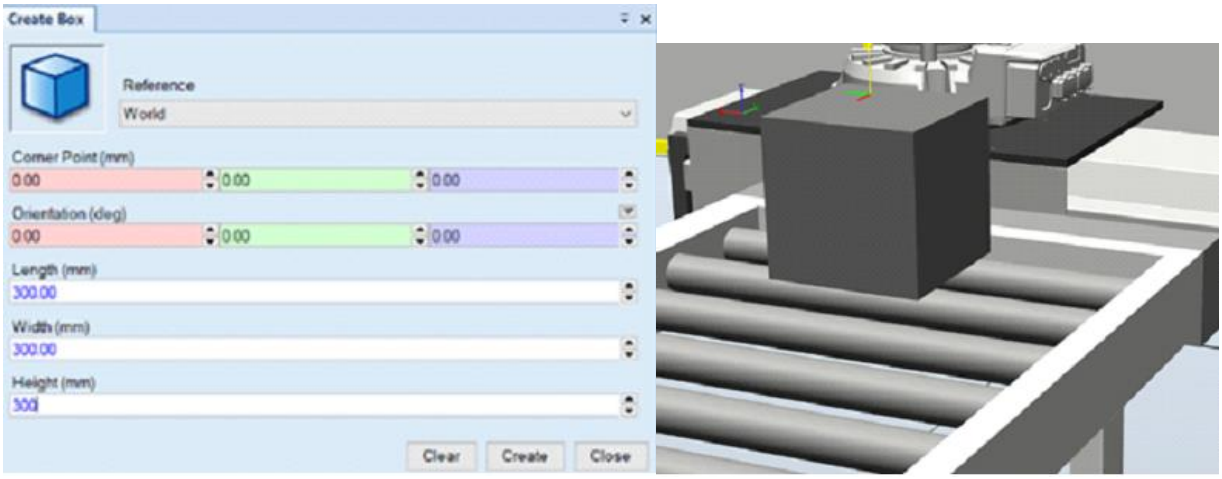


Fig. 6.23 Defining the manipulated object

The next step is to define the movements of the Industrial robot between the defined points Fig. 6.27.

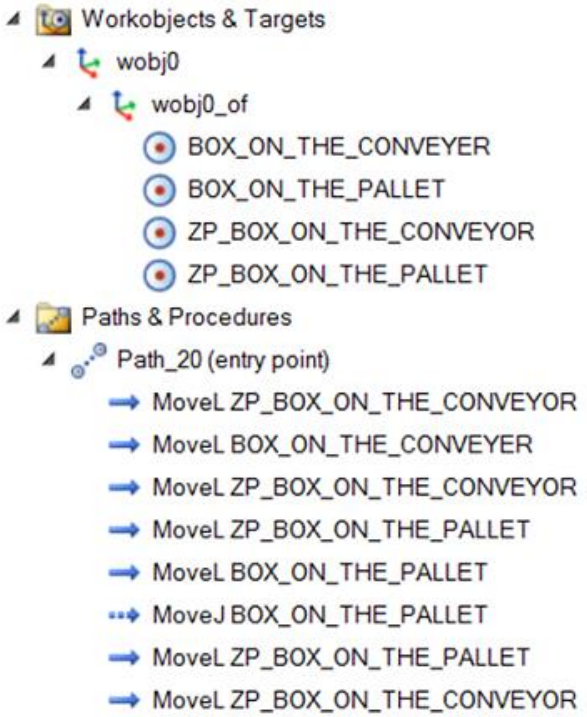


Fig. 6.27 Defining the movements of an industrial robot

The developed program can transfer string data from an external client written in a high-level language to the ABB RAPID server. The software option installed on ABB RobotStudio is "616-1 PC Interface" – required for socket communication.

Data transfer is shown in Fig. 6.30:

- data "string"/line 1 - VALUE in RAPID WATCH/ from an external client /Delphi/ to the server written in ABB RAPID
- data type – string, numbers from 0 to 9 and letters – Cyrillic letters – “А...Я” and- Latin letters – “A...Z”

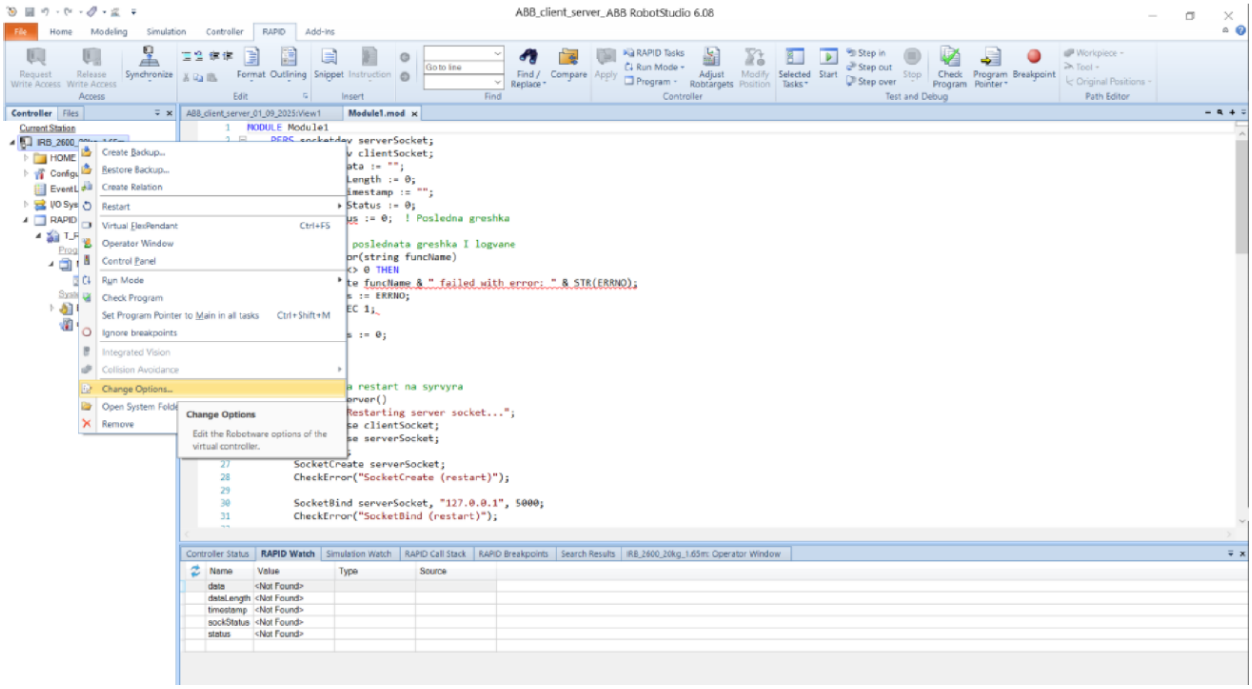


Fig. 6.30. Data transmission

6.3. Conclusions

The methodology developed in Chapter 4 has been used for programming industrial robots in the Fanuc ROBOGUIDE and ABB RobotStudio environments. The analysis of the results obtained proves its functionality.

Using Fanuc ROBOGUIDE, a server written in KAREL was developed, which can be used with a client program written in almost any high-level programming language such as C, C++, C#, Java, Python, PERL, Delphi, etc., complying with the Socket Programming syntax.

Using the high-level language Python, a client was created and tested to work with the server created in the Fanuc ROBOGUIDE environment.

Using ABB RobotStudio, RAPID output code was generated for programming industrial robots performing "Pick and Place" operations for conveyor service.

A server written in RAPID ABB was created, used for data transfer and operation of industrial robots.

CHAPTER 7 PROGRAMMING THE INDUSTRIAL ROBOT IN THE PRESENCE OF AN ADDITIONAL AXIS

7.1. Programming Industrial Robots with an Additional Axis

The use of an additional axis of motion, providing mobility to the Industrial Robot in order to expand its working area, is a common practice in the design of robotic cells. The control of this additional axis must be connected to the Industrial Robot controller so that it can be controlled by the program running on the Industrial Robot.

The presence of an additional axis of motion connected to the PR controller requires additional steps in the development of a Control Program for Industrial Robots using CARC systems.

7.2. Programming the PR in the presence of an additional axis in the middle of ROBOGUIDE

ROBOGUIDE provides users with various tools for developing control programs when an additional axis is available. To configure the additional axis in the ROBOGUIDE environment, the virtual mobile console of Fanuc LR Mate 200iD/7L will be used, which has the full functionality of the real one.

Using the virtual mobile console, we start a dialogue to add an additional axis Fig.7.3, by selecting

Maintenance - > Extended Axis Control - > F4(Manual) - >

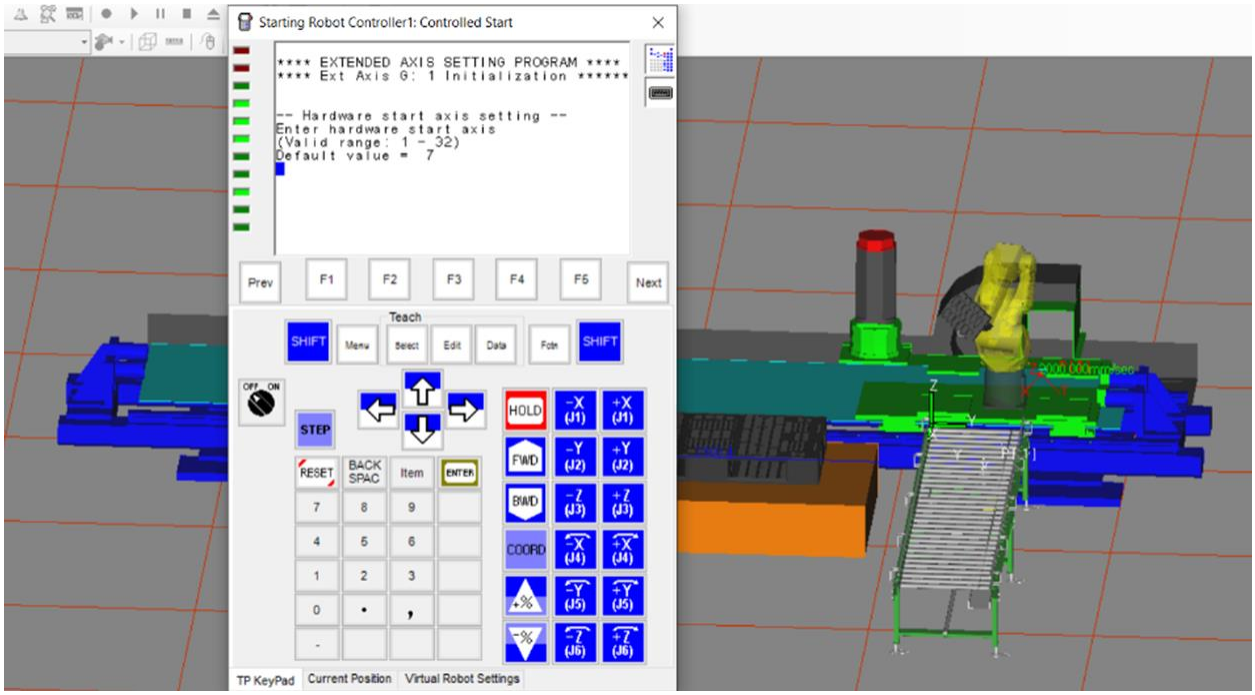


Fig. 7.3 Selecting the maintenance menu

7.3. Programming industrial robots with an Additional Axis in the middle of RobotStudio

To configure the additional axis in RobotStudio, the libraries with industrial equipment supplied with the system will be used.

3D models of robotic cells for performing "Pick and Place" operations using an additional axis are shown as follows:

- Fanuc LR Mate 200iD/7L industrial robots and an additional axis in the middle of Fanuc ROBOGUIDE - Fig. 7.16;
- ABB IRB 2600 industrial robots and IRBT-2005 lower movable rail in the middle of ABB RobotStudio - Fig. 7.17.

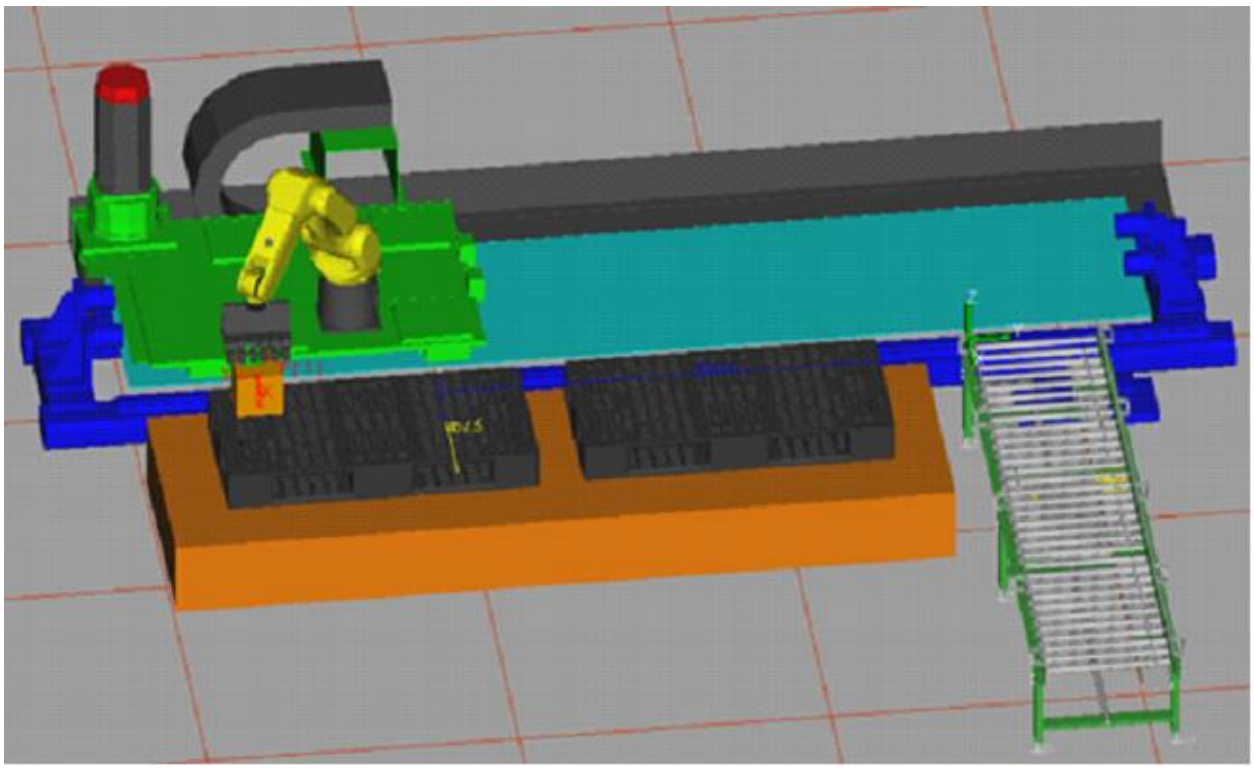


Fig. 7.16 Additional axis in the middle of FANUC ROBOGUIDE

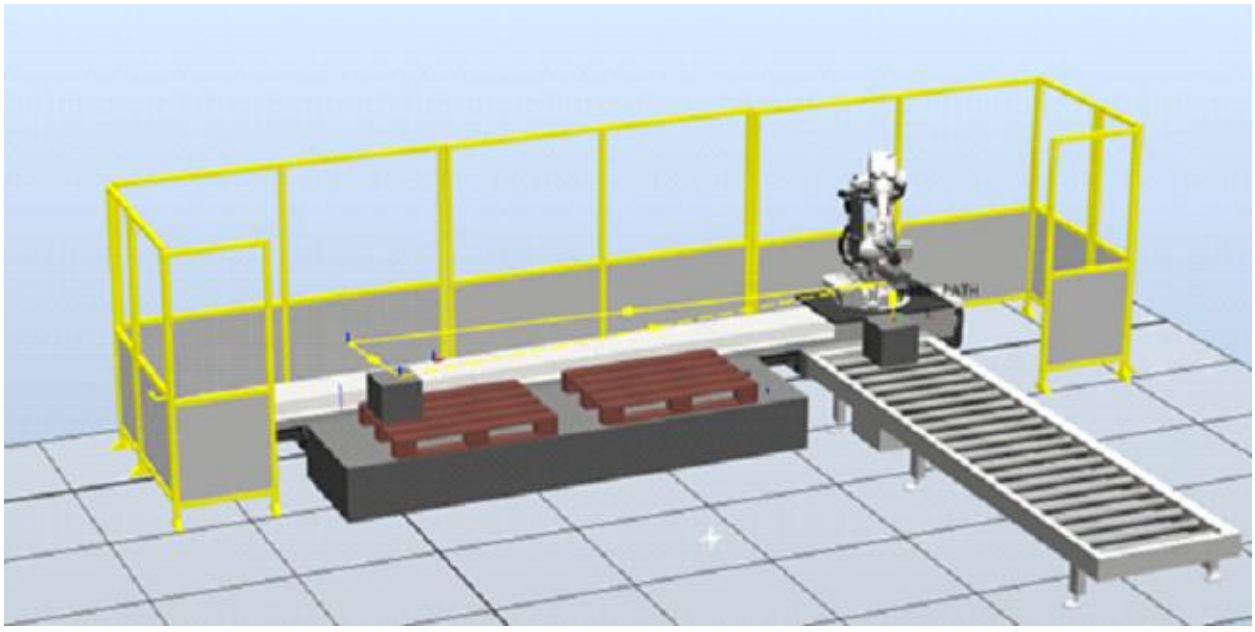


Fig. 7.17 Additional axis in the middle of ABB RobotStudio

7.4. Programming the PR in the presence of an additional axis using KAREL

Depending on the robot configuration, additional axes can be activated and configured in the controller via the corresponding system parameters.

KAREL, as a high-level programming language for FANUC robots, allows control of position data and logic for all axes, as well as the initiation of movements via TP programs.

By default, KAREL does not provide direct motion commands such as JMOVE or LMOVE. Direct control of motions and trajectories from KAREL is only possible when the Motion Option (J601) software option is activated.

Below is an example program from KAREL fig.7.18 for controlling the seventh axis:

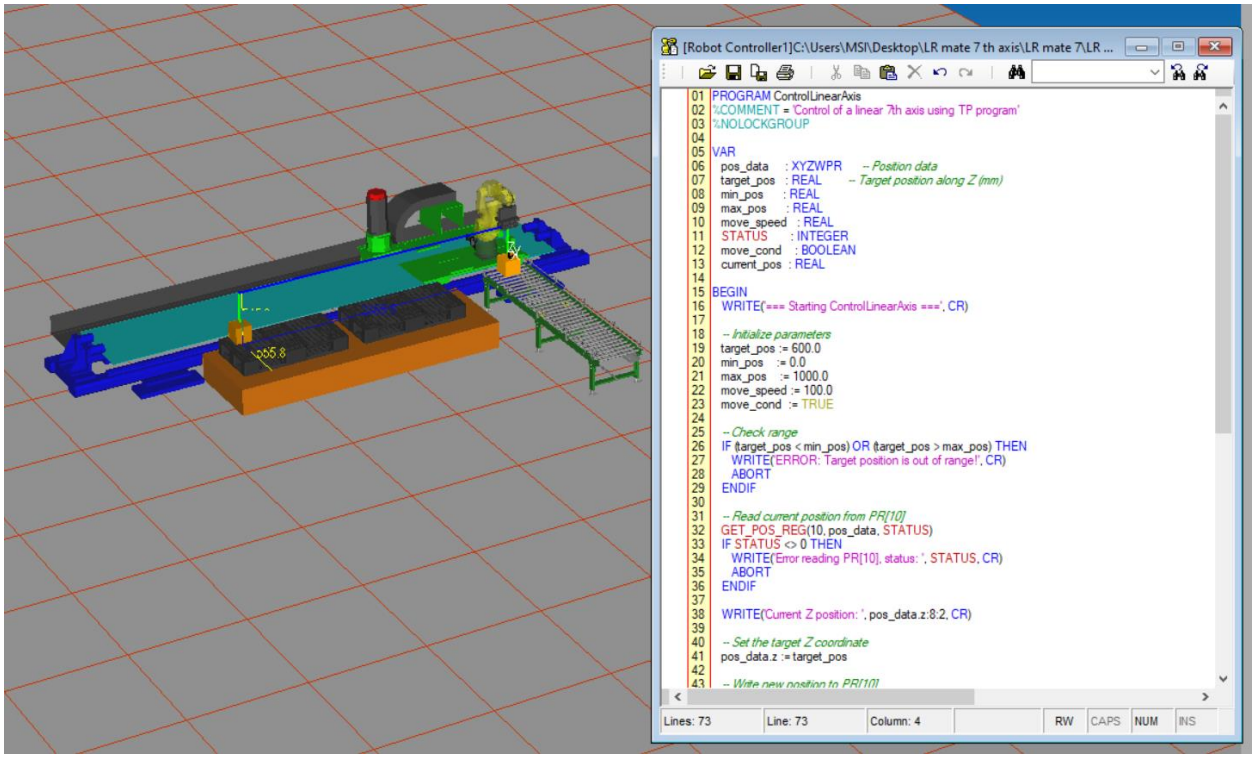


Fig. 7.18 KAREL program for controlling the seventh axis in the Fanuc Roboguide programming environment

7.5. Designing a robotic cell for performing "Pick and Place" operations in the ROBOGUIDE environment

With the increasing complexity of the operations performed by robotic systems and the development of computer technologies, CARC systems are becoming a standard tool in engineering practice. These systems are constantly evolving, and in addition to developing programs for industrial robots, they provide their users with the opportunity to integrate industrial robots into the structure of various robotic cells.

To demonstrate these capabilities, we will examine the design of a robotic cell for "Pick and Place" operations, consisting of palletizing boxes measuring 300x250x200 [mm] and weighing 5 [kg] in four rows on EUR 2 1200x1000 [mm]. Fig. 7.19 shows six possible configurations of the designed cell, taking into account the dimensions of the working area of the industrial robot used. The cell includes the following main components:

- Conveyor belt – delivers the boxes to the robotic palletizing cell;
- industrial robot - arranges the boxes on the pallets;
- pallets - define the area for arranging the boxes;
- additional axis - provides expansion of the working area of the Industrial Robot.

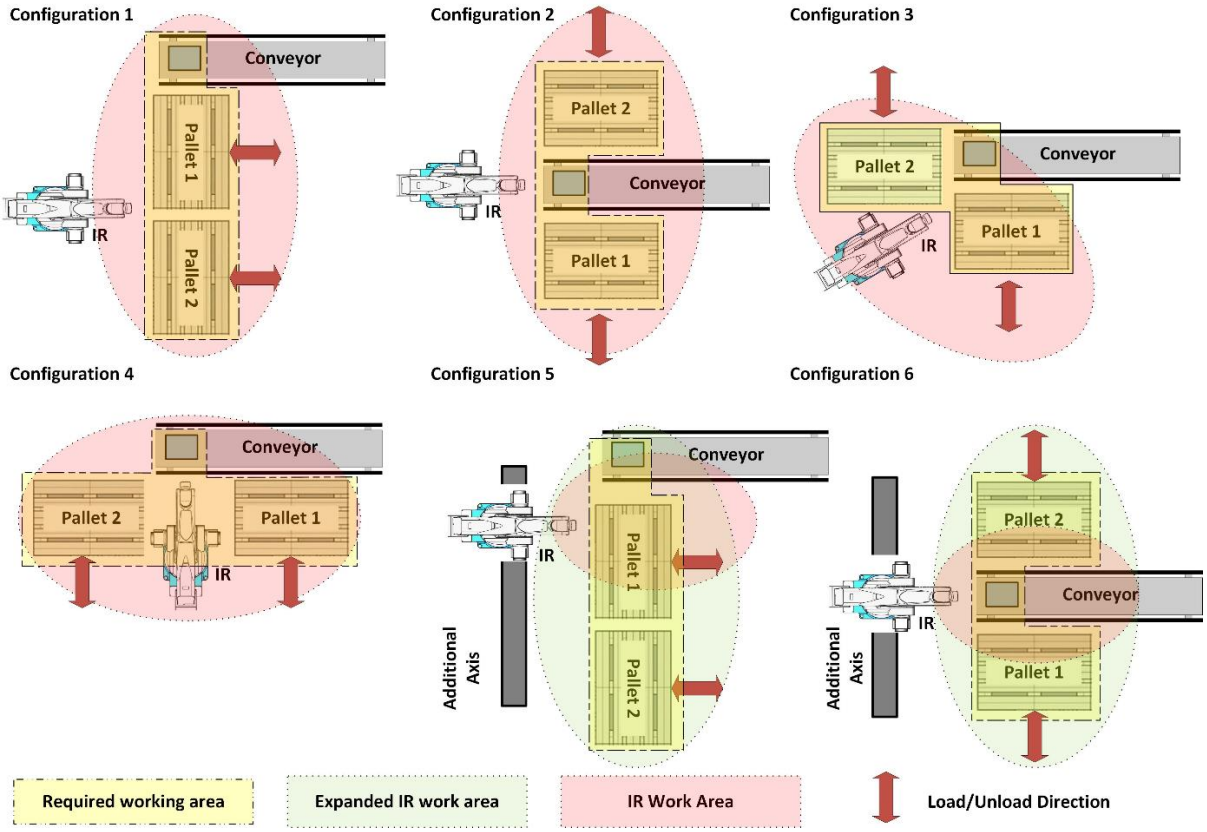


Fig. 7.19 Configurations of the designed cell

The analysis of the developed configurations includes an assessment of the capabilities of the selected industrial robot to perform the necessary operations at the specific location of the cell components. If necessary, the industrial robot can be replaced with another or an additional axis of mobility can be added in order to expand its working area.

The outcome of this step, based on the configurations shown in Fig. 7.19 and simulated using ROBOGUIDE, for the designed robotic cell, is presented in Fig. 7.22.

The analysis demonstrates that the maximum reach and workspace dimensions of the robots selected in step (3) are sufficient to carry out all required operations.

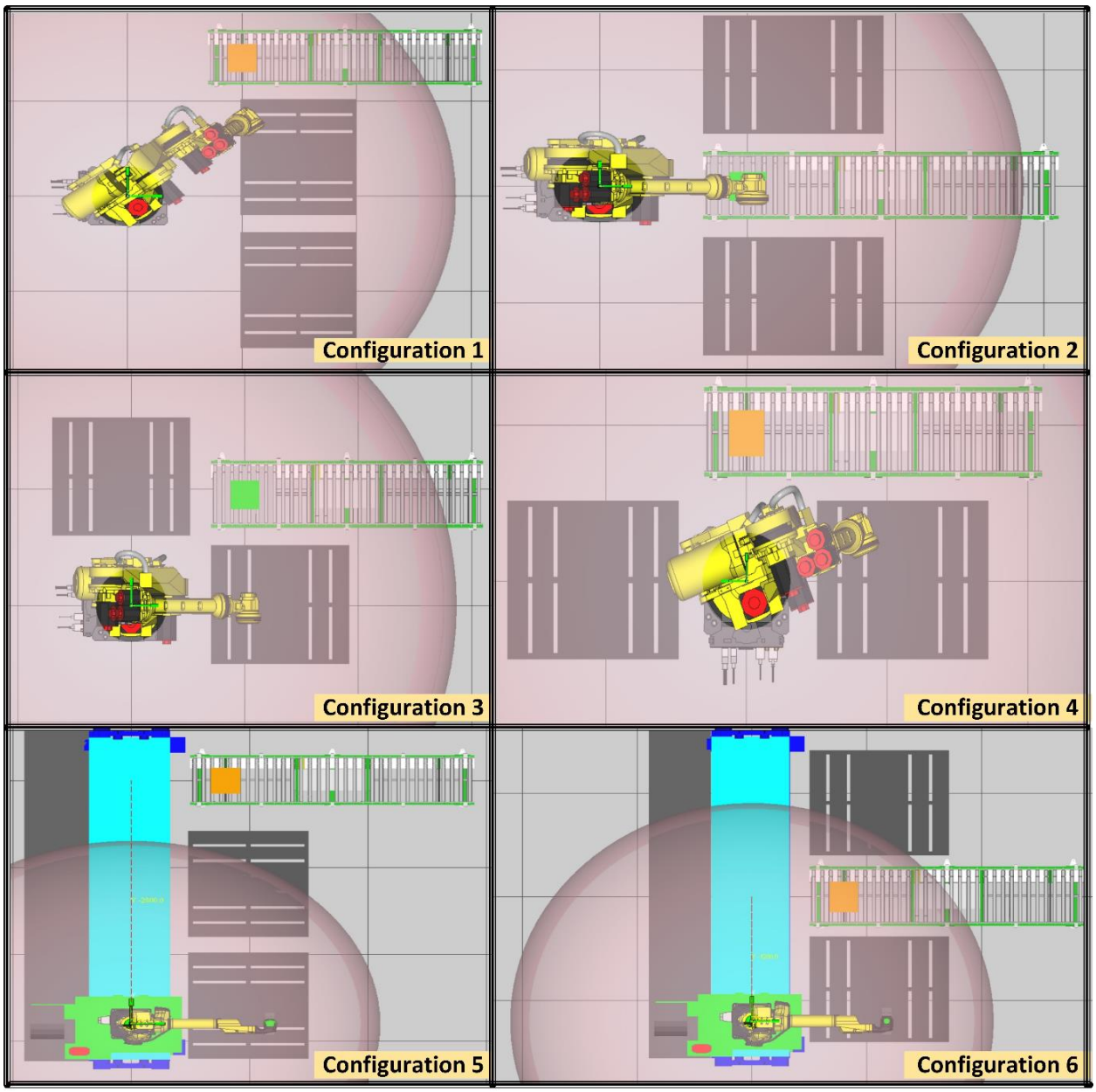


Fig. 7.22 Analysis of the developed configurations in the ROBOGUIDE environment

Table 7.2 Characteristics of the developed configurations

Configuration	Box arrangement time [min]			Total area [m²]	Cost
	Pallet 1	Pallet 2	Total		
1	7.58	10.91	18.49	7	1
2	7.58	7.58	15.16	8	1
3	7.58	7.48	15.06	6	0,92
4	10.78	10.78	21.56	9	0,93
5	4.86	6.20	11.06	11	0,77
6	4.86	4.86	9.72	11	0,77

The data in Table 7.2 show that, according to the established criteria:

- the configuration with the shortest box arrangement time on the two pallets (highest productivity) is **Configuration 6 – 9.72 [min]**;
- the configuration occupying the least area is **Configuration 1 – 7 [m²]**;
- the configurations with the lowest cost are **Configurations 5 and 6**.

Based on these data, **Configuration 6** is considered optimal and has been selected for use in the subsequent stages of the design process.

7.6. Conclusions

The use of an additional axis of motion provides mobility for the industrial robot, aiming to expand its workspace.

Additional steps have been defined in the development of control programs for industrial robots using CARC systems when an additional axis of motion is present.

Using a virtual mobile console on the Fanuc LR Mate 200iD/7L within the ROBOGUIDE environment, the configuration of the additional axis of motion was carried out.

The configuration of the industrial robot's additional axis of motion was also performed in the RobotStudio environment, using the libraries with industrial equipment provided by the system.

The main steps in designing robotic cells using CARC systems have been defined.

These steps were applied in the design of a robotic cell for executing "Pick and Place" operations within the ROBOGUIDE environment. As a result, six configurations of the designed cell were developed.

Using ROBOGUIDE, characteristics used to evaluate the configurations were determined. Based on the evaluation, an optimal variant of the designed cell was selected according to the assignment criteria.

The obtained results show that modern CARC systems are a powerful tool not only for programming industrial robots but also for designing robotic cells.

The defined steps are universal and can be applied when working with different CARC systems.

SCIENTIFIC-APPLIED AND APPLIED CONTRIBUTIONS

SCIENTIFIC APPLIED CONTRIBUTIONS

SAPC.1 A universal methodology has been developed for generating control programs for industrial robots using CARC systems, and the possibilities of using APIs in its implementation have been analyzed.

SAPC.2 A classification of types of attacks on robotic production systems has been developed according to three criteria, and recommendations for protection against these attacks have been provided..

APPLIED CONTRIBUTIONS

AC.1 A comparative analysis of the most commonly used high-level programming languages for APIs in industrial robot exploitation and their areas of application has been made.

AC.2 Server and client code for TCP/IP connection using FANUC KAREL and ABB RAPID robot programming languages has been developed and compared.

AC.3 Code for data encryption during industrial robot operation has been developed and analyzed, using various high-level programming languages supported by APIs in robot operation.

AC.4 A server written in KAREL has been developed, which can be used together with a client program written in different high-level programming languages supported by APIs in robot operation.

AC.5 The main steps in designing robotic cells using CARC systems have been defined; these steps are universal and can be applied when working with different CARC systems.

LIST OF PUBLICATIONS RELATED TO THE DISSERTATION

1. Vladimirov B., *Network Security in Remote Programming of Industrial Robots*, Automation of Discrete Production, No. 3, July 2021, pp. 171–175, ISSN 2682-9584
2. Vladimirov B., St. Nikolov, Sl. Dimitrov, *Application of Industrial Robots in Environments Hazardous to Human Health*, Automation of Discrete Production, No. 4, July 2022, pp. 94–98, ISSN 2682-9584
3. Vladimirov B., Nikolov St., Tsolov S., *Programming Industrial Robots in the Fanuc ROBOGUIDE Environment*, Engineering Proceedings, DOI: 10.3390/engproc2024070020, 2024 – Q4, Scopus
4. Vladimirov B., Nikolov St., *Remote Transmission of Information During the Operation of Fanuc Robots Using Fanuc ROBOGUIDE*, Annals of DAAAM and Proceedings of the International DAAAM Symposium, pp. 332–339, https://doi.org/10.2507/35th.daaam.proceedings.045, 2024 – Scopus
5. Nikolov St., Vladimirov B., *Designing Robotic Cells in the ROBOGUIDE Environment*, Proceedings of the 34th International Scientific and Technical Conference Automation of Discrete Production Engineering 2025, Az-buki National Publishing House, Bulgaria, https://doi.org/10.53656/adpe-2025.06

CITATIONS OF PUBLICATIONS RELATED TO THE DISSERTATION

Publication 3 is cited in:

1. Taj, S., Awasthi, S., Dahri, H., Hashsham, S., Khan, R., *The Evolution of Industrial Automation and Cybersecurity Risks*, Advancing Cybersecurity in Smart Factories through Autonomous Robotic Defenses Book Chapter, 2025, ISBN 979-833730585-1, 979-833730583-7, Pages 397 – 429, DOI: 10.4018/979-8-3373-0583-7.ch015

Publication 4 is cited in:

1. Sokolov, O., V., Andrusyshyn, A., Iakovets, V., Ivanov, *Intelligent Human–Robot Interaction Assistant for Collaborative Robots*, Electronics (Switzerland) Article Open Access, 2025 Multidisciplinary Digital Publishing Institute (MDPI), ISSN 20799292, Volume 14, Issue 6, Article number 1160, DOI: 10.3390/electronics14061160
2. Di, Gai, Xu, Weiyan, *Simulation Design of Industrial Robot Handling Workstation Based on ROBOGUIDE*, Lecture Notes in Electrical Engineering, Conference Volume 1441 LNEE, Pages 242 – 249, Book Series ISSN 18761100, Springer Science and Business Media Deutschland GmbH, 2025, ISBN 978-981968002-3, DOI: 10.1007/978-981-96-8003-0_27

Research project supporting the PhD candidate, National Institute of Research, Technical University – Sofia, 2024, Contract No. 241PD0024-06
Title: “Investigation of the Possibilities for Programming Industrial Robots Using APIs”
Supervisor: Prof. Eng. Stiliyan Nikolov Nikolov PhD
PhD Candidate: Eng. Boryan Chavdarov Vladimirov MSc

SUMMARY

PROGRAMMING OF INDUSTRIAL ROBOTS USING API

Eng. Boryan Chavdarov Vladimirov MSc

In the dissertation thesis, the possibilities provided by different computer systems for using API in the operation of IR (Industrial Robots) are investigated.

In the first chapter, a brief historical overview of the history of IR, control systems, and programming methods used in the operation of IR is made. The main functions of the application programming interface API (Application Programming Interface) are reviewed.

In the second chapter, high-level programming languages used in automated programming of IR are reviewed. A comparative analysis of the capabilities of the programming languages ST, ABB, RAPID and Fanuc KAREL is performed. The use of programming languages Fanuc Karel and ABB RAPID in automated programming of IR is analysed.

The third chapter examines the main communication protocols through which remote operation of IR can be achieved. Server and client code for TCP/IP connection are developed and compared using IR programming languages Fanuc KAREL and ABB RAPID.

A classification of the types of attacks against robotic production systems is developed, and recommendations for protection against these attacks are provided.

In the fourth chapter, a comparative analysis of the functional capabilities of various modern CARC systems is performed, based on ten criteria. A universal methodology, including eight stages, is developed for generating Control Programs (CPs) for IR using CARC systems.

The fifth chapter discusses the use of API in the operation of IR. Some of the main characteristics when using languages Fanuc KAREL, ABB RAPID, C++, C#, Python and PERL in the exploitation of IR are compared. Code for data encryption in the operation of IR, using various high-level programming languages used in the API during IR operation, is developed and analysed.

In Chapter 6, the methodology developed in Chapter 4 is used for programming IR in the environment of Fanuc ROBOGUIDE and ABB ROBOT STUDIO, and the obtained results are analysed.

In Chapter 7, the specifics of programming IR with an additional axis are discussed. The main steps in designing robotic cells using CARC systems are defined. These steps are applied to the design of a robotic cell for performing “Pick and Place” operations in the environment of Fanuc ROBOGUIDE.