Vpeljava SCADA sistema s pomočjo orodja Ignition na proizvodno linijo

Anej Marušič
FAMNIT, Koper
89162017@student.upr.si

***Povzetek* — Namen tega seminarja je podrobneje spoznati SCADA sistem. To smo naredili najprej z navedbo in kratko raziskavo zakaj pride do potreb za uporabo SCADA sistema. V pregledu področja smo se nato posvetili nekaterim možnostim za implementacijo takega sistema ter njihove prednosti in slabosti. Naposled smo izbrali eno od možnosti, s katero smo SCADA sistem implementirali. Implementacija SCADA sistema s pomočjo programskega orodja Ignition nam omogoči globlje razumevanje SCADA sistemov in njihove izgradnje, prav tako pa tudi izgradnje sistema na splošno in inkorporacije sistema v okolje v podjetju.**

Ključne besede — SCADA sistem, sistem, Ignition, Inductive Automation, LabView, sledljivost proizvodne linije

Key words — SCADA system, system, Ignition, Inductive Automation, LabView, traceability of the production line

I. UVOD

V sodobni industriji težimo k vedno večji avtomatizaciji in čim manjši odvisnosti proizvodnega cikla določenega izdelka od delavca na proizvodnji liniji. Kljub temu, da je na proizvodni liniji vedno manj ljudi zadolženih za določeno delovno mesto in so delavci primerno izobraženi o delu, ki ga opravljajo, še vedno prihaja do napak delavcev, zaradi katerih lahko izdelek s slabo kakovostjo pride do kupca. Podobno lahko pride tudi do napak, ki jih zakrivi naprava zaradi slabega delovanja (proizvajalec naprave namreč težko zagotovi stoodstotno zanesljivost).

Veliko od zgoraj omenjenih napak razrešujejo že naprave same z različnimi testiranji in tako sporočijo delavcu o neustrezni kvaliteti izdelka, vendar omenjeno velikokrat ni dovolj. Zgodi se lahko namreč, da delavec spregleda opozorilo in izdelek preda na naslednjo delovno postajo. Na enak način pa tudi v primeru, ko delavec stori pravilno odločitev omenjeno uniči ritem proizvodnje (delavec mora izdelek odstraniti in zapisati zakaj je bil izdelek odstranjen) in se tako podaljša cikel proizvodnje posameznega izdelka.

Iz omenjenega je precej razvidno zakaj je težnja, k čim večji avtomatizaciji proizvodnje vedno višja. K avtomatizaciji posameznega delovnega mesta na proizvodni liniji večino prispeva stroj sam, potrebujemo pa sistem, ki bo povezoval delovna mesta med sabo in ne bo prepuščal odločanja o ustrezno opravljeni operaciji na delovnem mestu samo delavcu samemu.

Na enak način pa nam veliko naprav na posameznih delovnih mestih nudi tudi veliko drugih podatkov, ki vplivajo na kakovost izdelka in nam tako lahko s primerno analizo le-teh na dolgi rok pomagajo priti do marsikaterih ugotovitev. Prav zato je lahko koristno, če se omenjene podatke shranjuje in naknadno analizira. S shranjevanjem teh podatkov bi lahko tako omogočili tudi boljši nadzor nad izmetom iz redne proizvodne linije brez nepotrebne obremenitve delavca z zapisovanjem razloga za izmet – kar bi med drugim pospešilo tudi proizvodni cikel.

Problem shranjevanja podatkov iz proizvodne linije in problem nadzora nad proizvodno linijo sta tako samo dve plati iste medalje. Obstaja veliko obstoječih rešitev, ki rešuje omenjena problema, zato se soočamo tudi s problematiko katero rešitev uporabiti. Izbira rešitve je seveda odvisna od veliko faktorjev znotraj podjetja – zahtev podjetja, razpoložljive delovne sile, finančne zmogljivosti in že utečenih praks ter sistemov v podjetju. Vsekakor pa je končen cilj pridobiti rešitev, ki bom čim bolj univerzalna in bo pripomogla, k čim večjemu dobičku podjetja [1].

Vse zgoraj omenjene probleme rešuje SCADA (*Supervisory Control And Data Acquistion*) sistem na proizvodni liniji [1].

Namen tega besedila je narediti kratek pregled področja in predstaviti zasnovo SCADA sistema in implementacijo le-tega z namenom podrobnejše raziskave implementacij SCADA sistemov.

II. PREGLED PODROČJA

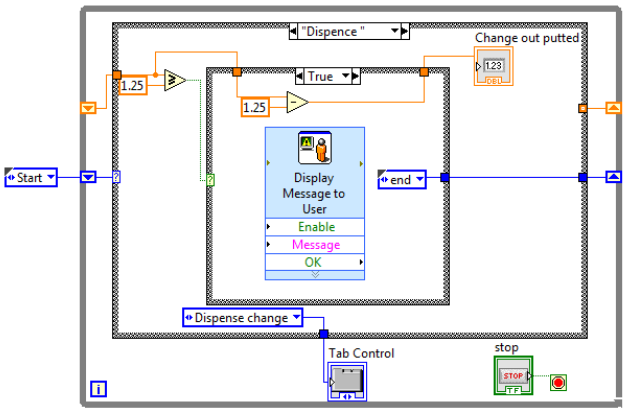
V tem razdelku bomo na kratko opisali opisali možnosti kako implementirati SCADA sistem na proizvodno linijo.

Pogosto podjetja za implementacijo uporabljajo eno od slednjih rešitev: uporaba visokonivojskega programskega jezika LabView, uporaba programskega orodja Ignition ali pa razvoj povsem lastne t. i. »hišne« rešitve. Zraven teh treh, katerim se bomo v tem članku podrobneje posvetili imamo še kar nekaj drugih rešitev. Med te rešitve spada uporaba raznih orodij kot so Siemensov WinCC Professional, Schneiderjev Wonderware InTouch, Rockwellov RSView, Automation studio podjetja B&R ter še mnogo drugih [2][3].

A. LabView

LabView je plačljiv visokonivojski programski jezik podjetja National Instruments. Je grafični programski jezik, kar pomeni, da za razliko od večine drugih programskih jezikov programov ne pišemo s pomočjo tipkanja programske kode, temveč s pomočjo vpeljave grafičnih komponent na ravnino ter implementiranja relacij med njimi. Primer izgleda enostavnega programa lahko vidite na sliki Slika 1. V prvoti je namen tega programskega jezika enostavno zajemanje podatkov iz naprav v laboratorijih z dodatnimi funkcionalnostmi. Kot grafični programski jezik je tako zgrajen predvsem z namenom, da je kar se da enostaven za uporabo neprogramerjem – znanstvenikom, ki hočejo le zajeti podatke iz naprav v laboratorijih ter morebiti še kakorkoli obdelati te podatke. Potemtakem sam po sebi torej ni namenjen implementaciji celotnega sistema, vendar pa polna in profesionalna različica LabView-ja omogočata vse potrebno za vpeljavo le-tega (komunikacija s PLC-ji naprave, pisanje in branje iz podatkovne baze, logične operacije in zanke, uporaba modulov) zatorej se velikokrat uporablja tudi v ta namen še posebej ko pride do primerov, da podjetje, ki skuša implementirati SCADA sistem noče oz. ne more zaposliti programerjev [4]. Licenca polne različice

LabView-ja znaša okoli 4000€, profesionalne pa 7000€ letno. Potrebno je pa poudariti, da lahko zgrajen program v tem programskem jeziku nato poganjamo brezplačno [5]. Problem programskega orodja LabView je v tem, ker v prvotni obliki ni namenjeno gradnji SCADA sistema in ima neko drugo namembnost. Prav tako je tu problem pridobiti kader za delo v omenjenim programskem jeziku saj je tukaj zahtevano precej specifično znanje – programerjem namreč grafično programiranje ni domače, neprogramerjem pa sploh programiranje ni domače.



Slika 1: Izgled enostavnega programa v programskem jeziku LabView [6]

B. Ignition

Ignition je programsko orodje v celoti namenjeno vpeljavi SCADA sistemov. Podjetje Inductive Automation ga je razvilo v okolju Java JDK in je zaradi tega zlahka prenosljivo med platformami. Omogoča enostavno komunikacijo s podatkovnimi bazami in PLC-ji naprav ter pisanje skript v programskem jeziku Python. Ignition je sicer licenčno orodje, katerega licenca za eno implementacijo za nekakšno osnovno različico, ki omogoča normalno delovanje znaša okoli 10.000$. Ker je orodje modularno imamo možnosti dokupa še raznih drugih modulov, ki lahko koristijo za lažjo implementacijo SCADA sistema neprogramerjem ali dodajo dodatne funkcionalnosti, kot je npr. alarmiranje [7].

Osnovne komponente Ignition orodja so tri: mrežni prehod, orodje za oblikovanje in odjemalec. Mrežni prehod tako služi kot spletni strežnik, kjer se izvajajo vsi glavni procesi, medtem ko odjemalec služi kot vmesnik uporabniku, ki lahko potem preko tega upravlja s sistemom. Orodje za oblikovanje pa je namenjeno le programerju, ki oblikuje uporabniški vmesnik ter sprogramira logiko sistema.

Enako kot LabView tudi Ignition zahteva neko bolj specifično znanje, vendar niti približno toliko kot grafično programiranje. Naučiti se moramo namreč le uporabiti orodje, vse programiranje pa poteka v znanem programskem jeziku Python.

C. Hišna rešitev

Lastna rešitev je precej težka za implementacijo, zato pa dopušča največ svobode. Tako kot vsako je potrebno tudi to rešitev učinkovito zastaviti in ustrezno načrtovati, vendar lahko to v tem primeru naredi le izkušeni programer. Dobra stran je, da v tem primeru nimamo nikakršnih omejitev (programska orodja imajo vedno nekatere omejitve), ter velikokrat precej malo stroškov z licenciranjem (kar je sicer odvisno od tega, če in kakšna orodja uporabljamo za programiranje), kar je tudi večkrat en od razlogov za uporabo te rešitve. Tipično hišna rešitev tudi zahteva delo precej več programerjev kot je to v primeru uporabe raznih orodij, saj je potrebno funkcionalnosti, ki jih npr. v primeru Ignitiona zagotavlja orodje tokrat sprogramirati sam.

Primer take hišne rešitve je implementacija back-enda (komunikacij z napravami, podatkovno bazo) v programskem jeziku Java v ogrodju (*framework*) Spring ter front-enda (logika delovanja sistema, uporabniški vmesnik, upravljanje s sistemom) v programskem jeziku Typescript s platformo Angular.

III. Specifikacija zahtev primera SCADA sistema

V korist podrobnejšega raziskovanja področja SCADA sistemov smo se odločili za implementacijo enega primera takega sistema. Implementacijo začnemo z zajemom oz. specifikacijo zahtev ter funkcionalnosti, ki jih želimo imeti v sistemu, ki ga gradimo. Na proizvodni liniji za vsako posamezno delovno mesto želimo:

1) Dovoliti polizdelku opravljanje operacije na trenutnem delovnem mestu ali opravljanje operacije zavrniti
2) Shraniti procesne parametre, ki so nastali tekom opravljanja operacije na tem delovnem mestu
3) Oceniti ali zabeležiti uspešnost opravljanja operacije na tem delovnem mestu
4) Zabeležiti delovno mesto na katerega naj polizdelek nadaljuje po opravljeni operaciji na tem delovnem mestu
5) Poskrbeti, da stroj pravilno deluje v odvisnosti od tipa izdelka na katerem opravlja operacijo

Zgornjih 5 zahtev je obveznih, opcijsko pa je zaželena še implementacija nekaterih zahtev, ki operatorju proizvodne linije olajšajo delo.

6) Pregledovanje dokumentacije tipa izdelka, ki je trenutno v izdelavi na delovnem mestu
7) Pregledovanje zgodovine trenutnega delovnega mesta
8) Pregledovanje zgodovine izdelka, ki je trenutno v obdelavi na delovnem mestu

IV. Načrtovanje primera SCADA sistema

Ko imamo zahteve določene se lahko lotimo načrtovanja, kako realizirati navedene zahteve. Zamisliti si je potrebno sistem, ki bo kar se da univerzalen in se ga bo dalo uporabiti zlahka tudi vnaprej na morebitnih novih proizvodnih linijah.

Prve štiri zahteve (od zahteve *1)* do zahteve *4)*) lahko obravnavamo skupaj kot nek način (protokol) opravljanja operacije na določenem izdelku na dotičnem delovnem mestu. Protokol izmenjave in beleženja podatkov tekom opravljanja operacije si sledi in je implementiran na slednji način v točno takem vrstnem redu:

1) Izdelek prispe na delovno mesto
2) S čitalcem kod se prebere njegova serijska številka

3) PLC naprave na delovnem mestu zahteva od sistema dovoljenje za delo

4) Preko serijske številke sistem v podatkovni bazi preveri primernost izdelka za opravljanje operacije na tem izdelku

5) V primeru, da je rezultat točke *4)* pozitiven potem sistem sporoči napravi, da lahko ta prične z delom na tem delovnem mestu ter v podatkovno bazo zapiše, da se je operacija na izdelku pričela, v nasprotnem primeru ji sporoči, da tega ne sme storiti in naj zaključi z operacijo

6) Naprava opravlja operacijo na izdelku

7) PLC naprave sporoči sistemu, da je ta zaključila z operacijo ter, da ima pripravljene procesne parametre, ki jih je zajel tekom operacije

8) Sistem v podatkovno bazo zabeleži, da je izdelek zaključil z operacijo ter zabeleži njegove procesne parametre, ki jih pridobi od PLC-ja naprave. Iz PLC-ja pridobi tudi končno kakovost izdelka po opravljeni operaciji in tudi to zabeleži v podatkovno bazo ter glede na ta podatek zapiše tudi na katero delovno mesto naj izdelek sedaj nadaljuje.

Z uveljavitvijo tega protokola lahko torej zagotovimo zadovoljiv nadzor nad procesom dela na delovnem mestu, iz njega pa sledi, da če hočemo celovito izdelati predlagan sistem potrebujemo dobro sodelovanje tudi s strani PLC programerjev naprav – le oni nam namreč lahko zagotovijo ustrezno komunikacijo s SCADA sistemom, ki je navedena v protokolu.

Tako smo zadovoljili prve štiri zahteve. Pri zadovoljevanju zahteve *5)* lahko hitro ugotovimo, da potrebujemo še zunanji podatek izven sistema preko katerega pridobimo informacijo o tipu izdelka, ter posledično seveda tudi informacijo o tem, kako naj naprava opravlja operacijo na nekem tipu izdelka. Ta zunanji podatek lahko pridobimo preko MES sistema podjetja in le-ta ga pridobi preko ERP v podjetju. Potemtakem sledi, da je nujno inkorporiranje (oz. komunikacija) tudi MES sistema s SCADA sistemom. Ta nam mora priskrbeti:

- Nalog za izdelavo izdelka na določenem delovnem mestu (oz. naloge med katerimi lahko potem operator proizvodne linije izbira)
- Preko naloga tip izdelka, ki se izdeluje
- Preko tipa izdelka t. i. recepte za delovanje naprav (to so navodila za delovanje v dogovorjeni obliki PLC-jem naprav)

Potrebno je torej, da pred izvedbo našega osnovnega protokola izvedemo še slednjo sekvenco:

1) Sistem preveri, če je operater linije že izbral nalog na trenutnem delovnem mestu

2) V primeru, da je rezultat iz točke *1)* pozitiven, potem se lahko začne izvajati osnovni protokol, kajti naprava že ima ustrezna navodila za delo na izdelku, v nasprotnem primeru pa je potrebno preiti na točko *3)*

3) SCADA sistem zahteva od operaterja linije izbiro naloga za delo na izdelku

4) Preko naloga izbranega v točki *3)* SCADA sistem pridobi tudi tip izdelka in recepte ter jih posreduje PLC-ju naprave, kateri potem uskladi nastavitve naprave glede na dostavljen recept

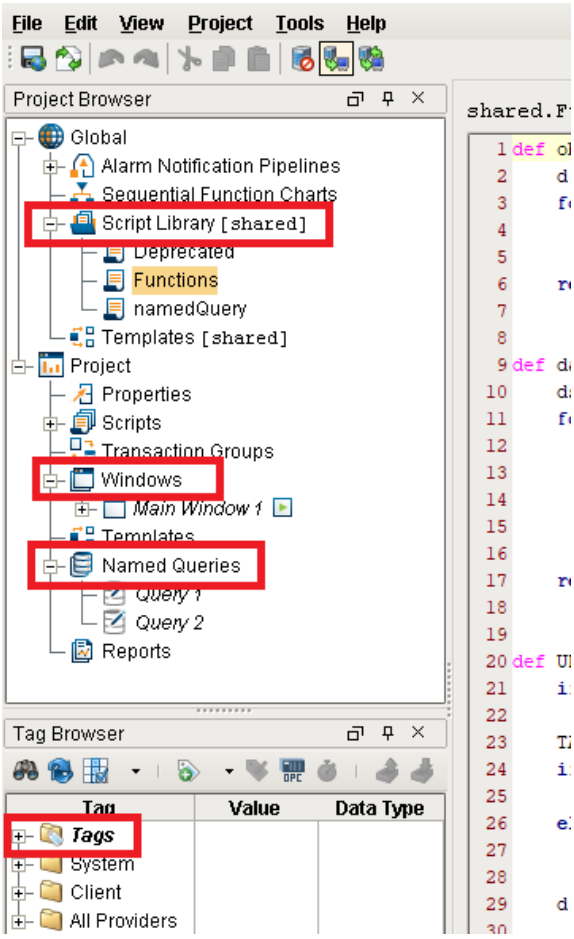
Opcijske zahteve, ki so namenjene zgolj operatorju proizvodne linije bi lahko izpustili saj nimajo pomena oz. utemeljenih primerov uporabe v primeru idealnega delovanja proizvodne linije. V takem primeru namreč sploh ne bi bilo razloga za zanimanje o zgodovini izdelka ali zgodovini delovnega mesta, če je bilo vse izvedeno tako kot je potrebno. Žal pa temu ni tako, okvare na napravah in materialih so redni neizogibni dogodki, prav tako pa pride tudi do napak delavcev in jih zato moramo neprestano dobro informirati preko dokumentacije. Implementacija teh treh opcijskih zahtev ne predstavlja nobenega velikega načrtovalskega izziva. Za točko *6)* potrebujemo namreč, le pridobiti ustrezno dokumentacijo s pomočjo MES sistema, za točki *7)* in *8)* pa le opraviti ustrezno poizvedbo po podatkovni bazi in rezultate le-te prikazati na želenem prikazovalniku.

V. Izdelava primera SCADA sistema

Za izdelavo primera SCADA sistema smo se odločili uporabiti programsko orodje Ignition. Od v II. poglavju naštetih je namreč to najbolj primerno, saj uporablja programski jezik, ki je programerjem domač (za razliko od LabView, ki zahteva zelo specifično znanje za programiranje v le-tem), zahteva manj dela s strani programerjev (za tak prototipni primer sta eden do dva programerja dovolj) ter omogoča brezplačno licenco, če se le-to podaljšuje vsake 2 uri, kar je za takšno testno izgradnjo dovolj.

V tem poglavju bomo potemtakem skušali na čim bolj jedrnat in kratek način prikazati glavne povezave med komponentami ter opisati kako načrtovani SCADA sistem iz poglavja IV. implementirali v Ignition-u. Za to je potrebno, da na kratko opišemo nekatere komponente Ignition-a ki so bile uporabljene in katerih uporaba je esencialna.

Na sliki Slika 2 lahko vidimo izgled brskalnikov po komponentah programskega orodja. Na voljo imamo knjižnico skript (*Script Library*) kamor lahko hranimo skripte v katerih so funkcije, ki jih lahko kličemo od kjerkoli iz programskega orodja. V razdelku »Windows« lahko kreiramo nova okna, ki jih lahko prikazujemo na zaslonih ter ta okna urejamo s pomočjo uporabniškega vmesnika. »Named queries« komponenta nam omogoča kreiranje t. i. imenovanih poizvedb po podatkovni bazi s pomočjo programskega jezika SQL. Poizvedbo imamo tako že v naprej pripravljeno s parametri, ki jih posredujemo tej imenovani poizvedbi ob klicu le-te. Te poizvedbe lahko prav tako kot skripte kličemo od kjerkoli iz programskega orodja (tudi iz skript). Naslednja stvar, ki jo je tudi potrebno razumeti so tagi oz. značke. To so kot nekakšne globalne spremenljivke v programskem orodju, ki ohranjajo vrednost vedno, ne glede na to ali pride do kakršnihkoli napak v izvajanju ali do izpada električne energije. V te značke se preslikajo tudi značke PLC naprav na katere smo povezani s programskim orodjem, v značke pa lahko tudi pišemo s pomočjo skript ali ročno. Povezave s PLC napravami in podatkovno bazo so opravljene in specificirane preko mrežnega prehoda.



Slika 2: Izgled brskalnikov po komponentah programskega orodja Ignition

SCADA sistem smo implementirali z namenom, da ima vsako delovno mesto svoj prikazovalnik na katerem prikazujemo stanje naprave oz. v katerem koraku osnovnega protokola se operacija na trenutnem delovnem mestu nahaja. Torej smo za vsako delovno mesto ustvarili novo okno v Ignitionu, da lahko na vsakem oknu izpišemo korake protokola za tisto delovno mesto. Na teh oknih imamo tudi tri gumbe za: pregledovanje dokumentacije, pregledovanje zgodovine izdelka ter pregledovanje zgodovine delovnega mesta. Do dokumentacije dostopamo preko URL-ja preko web browser komponente, ki jo omogoča Ignition. Pregledovanje zgodovine izdelka in delovnega mesta pa smo aplicirali tako, da smo napisali dve imenovani poizvedbi, za vsako funkcijo svojo ter njuno izvedbo sprožili ob pritisku na gumb, ki odpre pogled zgodovine izdelka oz. zgodovine delovnega mesta

Pri izvedbi glavnega protokola smo postopali malce drugače. Ignition omogoča izvajanje skript v primerih, ko se vrednost neke značke spremeni. PLC tako preko značke sporoči serijsko številko katero je odčital na delovnem mestu ter sporoči da zahteva dovoljenje za delo. Ko se ta značka, ki zahteva dovoljenje za delo spremeni pokličemo funkcijo iz skripte iz knjižnice skript, ki ugotovi, da PLC naprave zahteva dovoljenje za delo, pridobi serijsko številko ter s temi podatki kliče imenovano poizvedbo katera naredi poizvedbo po tabeli v bazi in vrne delovno mesto na katerega mora ta serijska številka nadaljevati. Če klicana funkcija ugotovi, da je pridobljeno delovno mesto enako delovnemu mestu na katerem se izvaja skripta potem s klicanjem neke druge imenovane poizvedbe naredi v enaki tabeli kot prej nov zapis, kjer je zapisano, da je izdelek trenutno v obdelovanju na trenutnem delovnem mestu. Nato s tem, da vpiše v dogovorjeno značko, ki je slična s značko PLC-ja, vrednost 1 dovoli napravi začetek operacije. V nasprotnem primeru v to značko vpiše 2.

Podobno nato, ko naprava zaključi z operacijo to sporoči preko spremembe neke značke. SCADA sistem nato ponovno kliče funkcijo (tokrat drugo), ki prebere vrednosti procesnih parametrov iz značk, ki preslikujejo značke PLC-ja naprave in jih nato s klicem imenovane poizvedbe zapiše v drugo tabelo v podatkovni bazi. Nato enako s pomočjo neke druge značke funkcija v izvajanju ugotovi ali je bil polizdelek na delovnem mestu ustrezno dokončan ali morda ne. V obeh primerih se pokliče spet neka druga imenovana poizvedba katera posodobi tisti vnos v tabelo, ki je bil kreiran ob začetku operacije na tem izdelku. Posodobi ga v odvisnosti od kvalitete zaključka te operacije – v primeru, da je izdelek uspešno zaključil operacijo v stolpec kvaliteta zapiše 1 in zapiše, da je naslednje delovno mesto na katerega mora nadaljevati ta izdelek tisto delovno mesto, ki po načrtu proizvodne linije sledi trenutnemu delovnemu mestu. V primeru pa da je izdelek neuspešno zaključil operacijo se v stolpec kvaliteta zapiše 9 in kot naslednje delovno mesto, postaja reparature izdelka.

Tekom celotne izvedbe glavnega protokola v funkcijah skript delamo tudi izpise, ki se izpisujejo na oknu na delovnem mestu, da delavca informirajo kje se izdelek nahaja v sekvenci delovanja.

Na oknu na delovnem mestu imamo naveden tudi trenutni nalog pod katerim se izdelek izdeluje. V primeru, da tega naloga ni navedenega, delavec oz. naprava ne more začeti z operacijo dokler delavec ne pritisne gumba za izbiro naloga in v novem oknu izbere delovni nalog za izdelek. Razpisane delovne naloge zapiše MES sistem v podatkovno bazo in SCADA sistem nato naredi poizvedbo po bazi s pomočjo imenovanih poizvedb ob kliku na ta gumb za prikaz razpisanih nalogov.

Ob izbiri naloga se sproži nova funkcija v eni od skript, ki najprej spet sproži imenovano poizvedbo, ki preko naloga oz. tipa izdelka, ki je zapisan v nalogu izbere ustrezen recept, ki ga prav tako MES prinese do podatkovne baze. Ko Pythonova funkcija pridobi ustrezen recept v .json obliki, mora iz te .json oblike pridobiti ustrezne informacije o vrednosti nastavitvenih značk PLC-ja naprave (to so značke PLC-ja, ki so zopet preslikane v Ignition sistem). To se naredi s klicem druge funkcije, ki smo jo implementirali in katera izlušči omenjene podatke iz .json oblike. Te podatke nato ena od funkcij v Pythonu vpiše v ustrezne značke in zaključi z obdelavo receptov in dovoli PLC-ju naprave, da lahko ta začne z začetkom glavnega protokola.

VI. Zaključek

Raziskava opisana v tem članku je bila opravljena z namenom bližjega spoznavanja s SCADA sistemi. Na začetku smo se posvetili potrebi po uvajanju SCADA sistema sploh, nato pa smo prešli v spoznavanje možnosti za implementacijo. Možnosti na kakšen način pristopati, k implementaciji SCADA sistemov je veliko in le-te so si med seboj precej različne. Idealno bi bilo zgraditi sistem v vsaki od možnosti in potem te rešitve primerjati med sabo ter tako

temeljiteje spoznati implementiranje SCADA sistema ter s pomočjo primerjanja med dobljenimi rešitvami pridobiti najbolj ustrezno rešitev za dani problem. Žal pa je to prevelik projekt za razmere raziskovalnega seminarja in smo se zato lotili le implementacije SCADA sistema s pomočjo orodja Ignition.

Tekom implementacije smo se pobližje spoznali in raziskali proces nastajanja celotnega sistema. Spoznali smo pomembnost temeljite analize zahtev ter prišli do sklepa, da načrtovanje sistema zahteva med drugim tudi veliko sodelovanja z ekipami, ki v sistem posredujejo vhodne podatke. Kar torej pomeni veliko komunikacije s programerji PLC naprav ter programerji MES sistema v podjetju. Prav tako smo se še pobližje soočili s tem, da je za učinkovito implementacijo pomembno dobro poznavanje orodja katerega uporabljamo, kar še bolj potrdi sum o tem, da bi težko implementirali ta SCADA sistem z več različnimi orodji. Implementacija v Ignition programskem orodju nam ja pa sicer prinesla zadovoljiv rezultat – zaradi modularnega pristopa in prednosti orodja sklepamo, da bo ponovna implementacija tega SCADA sistema na neki novi proizvodni liniji veliko lažja. Kljub temu pa se zavedamo, da je to le prva različica oz. prototip SCADA sistema, ki se bo skozi čas izpopolnil – nadejamo se lahko predvsem pohitritve delovanja, ki je v proizvodnji bistvenega pomena.

VIRI

[1] A. Daneels, W. Salter, What is SCADA?, CERN, Geneva, Switzerland, 1999

[2] Bayt, What are the best SCADA software for 2015 considering price and efficiency?, 2015. [Elektronski]. Dostopno na: https://specialties.bayt.com/en/specialties/q/206447/what-are-the-best-scada-software-for-2015-considering-price-and-efficiency/

[3] Qoura, Which PLC and SCADA software is commonly used in industries?, 2018, [Elektronski]. Dostopno na: https://www.quora.com/Which-PLC-and-SCADA-software-is-commonly-used-in-industries

[4] Viewpoint Systems, LabView Uses – What is LabView used for?, [Elektronski]. Dostopno na: https://www.viewpointusa.com/labview/what-is-labview-used-for/

[5] National Instruments, LabView 2019, 2019, [Elektronski]. Dostopno na: http://www.ni.com/sl-si/shop/labview/labview-details.html

[6] Wikimedia, LabVIEW State Machine example (Dispense Case), 2014, [Elektronski]. Dostopno na: https://upload.wikimedia.org/wikipedia/en/b/ba/LabVIEW_State_Machine_example_%28Dispense_Case%29.png

[7] Inductive Automation, Ignition 8, 2019, [Elektronski]. Dostopno na: https://inductiveautomation.com/