

VIZUALIZACIJA AUTOSAR LOGGING/TRACING PROTOKOLA VISUALIZATION OF AUTOSAR LOGGING/TRACING PROTOCOL

Mladen Planojević, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – U ovom radu obrađeno je jedno rješenje alata za vizualizaciju AUTOSAR logging/tracing protokola. Rješenje omogućava prikaz izvršavanja izvršnih jedinica u okviru sistema u realnom vremenu koji je baziran na AUTOSAR arhitekturi, na osnovu čega je moguće izvršiti analizu i poboljšanje performansi sistema.

Ključne reči: Vizualizacija, AUTOSAR standard, TTEthernet protokol, logging, tracing

Abstract – This paper presents one of potential solutions for implementation of tool for visualisation of AUTOSAR logging and tracing protocol. The tool implements a graphical view of runnables execution within real-time system, based on AUTOSAR architecture. The visualization, generated by tool, helps in analysis conduction and improvement of system performance.

Keywords: Visualization, AUTOSAR standard, TTEthernet protocol, logging, tracing

1. UVOD

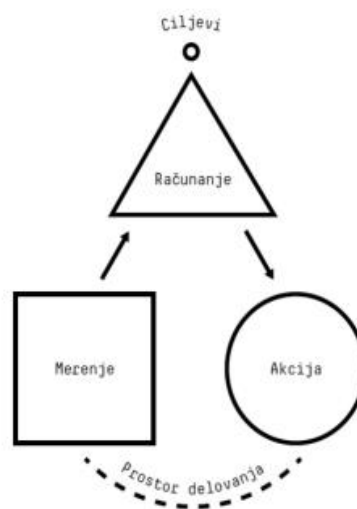
Automobilska industrija danas predstavlja jednu od najperspektivnijih grana industrije. Od svog nastanka, pretrpjela je brojne izmjene, a sve u cilju poboljšanja samog kvaliteta automobila, a time i postizanja sve većeg zadovoljstva putnika, kao krajnjih korisnika. Automobili današnjice predstavljaju veoma kompleksne sisteme, sastavljene od velikog broja komponenti, čiji su zadaci različiti – od pomoći vozaču pri svakodnevnoj vožnji pa sve do potpune autonomnosti i isključivanja vozača iz procesa upravljanja vozilom [1].

Rad pomenutih komponenti automobila zasniva se na principu takozvane kontrolne petlje. Osnovna kontrolna petlja, prikazana na slici 1, sastoji se od senzorskog sistema za prikupljanje informacija o sredini – mjerenje, računarskog sistema za donošenje odluka – računanje, kao i aktuatorskog sistema koji vrši uticaj nad sredinom – akcija [2]. Senzori, poput kamera, radara i lidara igraju ključnu ulogu u realizaciji kontrolne petlje, prikupljajući bitne podatke o različitim parametrima vozila, kao što su brzina, temperatura, pritisak, i slično. Vrijednosti pomenutih parametara se, zatim, obrađuju od strane moćnih procesnih jedinica kako bi se donijele odgovarajuće odluke i izvršile pravovremene akcije posredstvom aktuatora.

NAPOMENA:

Ovaj rad proistekao je iz master rada čiji mentor je bio dr Bogdan Pavković, vanr. prof.

Dati proces se odvija neprestano, tokom trajanja cijele vožnje i u realnom vremenu, a razmjena podataka se ostvaruje posredstvom komunikacionih medijuma i protokola.



Slika 1. Osnovna kontrolna petlja [1]

Da bi sastavni elementi kontrolne petlje, prikazane na slici 1, koordinisano i efikasno funkcionisali, potrebno je implementirati pouzdan i efikasan komunikacioni sistem, koji će raditi u realnom vremenu i čija će jedna od glavnih odlika biti determinizam [3]. Pojam determinizma, koji se veoma često sreće u sistemima koji se izvršavaju u realnom vremenu, unosi predvidivost u takve sisteme, odnosno sugerise da se sve aktivnosti, tj. događaji u datom sistemu odvijaju po tačno definisanom rasporedu i u predefinisanoj redoslijedu [4].

Upravo je determinizam jedna od glavnih odlika koja je presudila da TTEthernet protokol postane opšte prihvaćena tehnologija u automobilske industriji [5], a sistem na kom je baziran ovaj rad svoje izvršavanje zasniva na pomenutom protokolu.

Osim determinizma, kao ključnog faktora u komunikacionom sistemu automobila, bitno je pomenuti i AUTOSAR koncept (eng. *Automotive Open System Architecture*), koji pruža standardizovani okvir za razvoj softvera u automobilima. Pomenuti koncept, kreiran od strane istoimenog konzorcijuma, predstavlja globalno priznati standard u automobilske industriji, koji danas u svojim rješenjima koriste neki od najpoznatijih proizvođača automobila i koji je nastao kao odgovor na potrebu da se definiše otvorena i standardizovana arhitektura softvera koja bi omogućila lakšu integraciju različitih

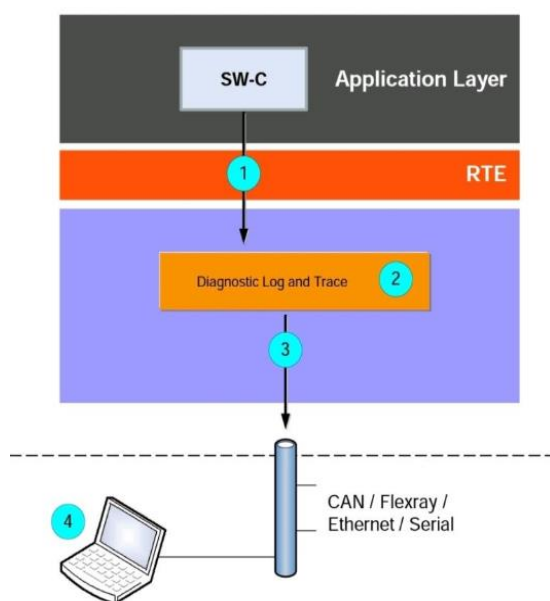
komponenti [6], kao i ponovnu upotrebu softverskih resursa među različitim proizvođačima i modelima automobila [7]. Ono što *AUTOSAR* standard donosi sa sobom kao novitet u automobilsku industriju jeste slojevita arhitektura, koja omogućava razdvajanje različitih aspekata razvoja automobilskih sistema [8], a glavni koncepti na kojima se bazira pomenuta arhitektura jesu:

- softverska komponenta (eng. *SW-C*, *Software Component*),
- virtualna funkcijska sabirnica (eng. *VBF*),
- sloj apstrakcije softvera (eng. *RTE*, *Run Time Environment*),
- bazični softver (eng. *BSW*, *Basic Software*) i
- izvršna jedinica (eng. *Runnable*).

U sistemima čije se izvršavanje odvija u realnom vremenu i gdje je podrazumijevano determinističko ponašanje, od ključne važnosti je postojanje rutina za praćenje performansi i aktivnosti softvera, a ono što, između ostalog, *AUTOSAR* koncept nudi jesu mehanizmi za logovanje događaja (eng. *logging*), snimanje tragova izvršavanja (eng. *tracing*), te prikupljanje dijagnostičkih informacija koje mogu biti od izuzetne pomoći, pogotovo u slučajevima kad je potrebno otkriti problem u kompleksnim automobilskim sistemima [9].

AUTOSAR logging/tracing protokol omogućava inženjerima da efikasno prate i analiziraju ponašanje softvera u realnom vremenu, a takođe ga je moguće jednostavno povezati sa alatima za analizu logova i dijagnostiku grešaka, što doprinosi poboljšanju performansi, pouzdanosti, te održivosti softverskih sistema u automobilu [10]. Sam koncept protokola i njegov položaj unutar slojevite arhitekture *AUTOSAR* standarda prikazani su na slici 2.

U praksi postoje pojedini alati za prikaz izvršavanja komponenti softvera, odnosno vizualizaciju *AUTOSAR logging/tracing* podataka, poput *Vector CANoe/CANalyzer* ili *Trace.check* alata.



Slika 2. Pozicija *logging/tracing* protokola u okviru *AUTOSAR* arhitekture

Međutim, dobavljanje licence za pomenuta, komercijalna rješenja je skupo i često neisplativo, a u određenim situacijama izostaje i podrška za dijelove implementacije rješenja, koji su specifični za određeni projekat. Upravo je to poslužilo kao motivacija za razvoj alata za vizualizaciju *AUTOSAR logging/tracing* protokola, čiji će koncepti i programsko rješenje biti objašnjeni u nastavku rada.

2. KONCEPT RJEŠENJA

Ovo poglavlje opisuje koncept rješenja za vizualizaciju *AUTOSAR logging/tracing* protokola. Dat je opis konceptualnih i tehničkih aspekata razvoja, odnosno teorijski prikaz koraka koje je neophodno preduzeti, počev od prikupljanja mjernih rezultata, preko obrade prikupljenih podataka, kako bi se na kraju došlo do rezultata, što je u ovom slučaju prikaz izvršavanja funkcionalnosti sistema u realnom vremenu, na osnovu čega je moguće veoma lako procijeniti stanje i unaprijediti performanse sistema.

• Postavka problema prikupljanja i vizualizacije podataka

Kako je već navedeno u uvodnom dijelu poglavlja, alat za vizualizaciju *AUTOSAR logging/tracing* protokola, predstavlja rješenje višestruke namjene. Primarni cilj alata je vizuelni prikaz redoslijeda i vremena izvršavanja izvršnih jedinica u sklopu automobilskog softvera, ali osim toga rješenje ima bitnu ulogu u unapređenju procesa otklanjanja grešaka u složenom sistemu, a samim tim i poboljšanje samog softvera. Da bi navedeno bilo realizovano potrebno je da alat ispuni sljedeće zahtjeve:

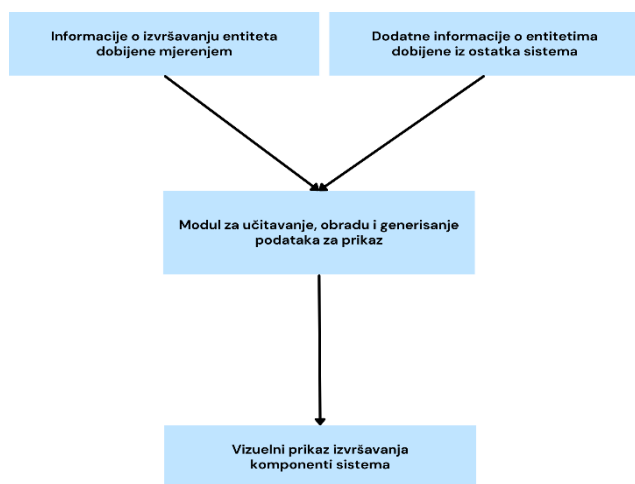
1. učitavanje mjernih podataka i prikaz izvršavanja izvršnih jedinica unutar jednog ciklusa,
2. pronalaženje i obilježavanje ciklusa čije izvršavanje potencijalno narušava stabilnost sistema i
3. pronalaženje i obilježavanje izvršne jedinice unutar nestabilnog ciklusa, čije izvršavanje narušava performanse sistema:
 - a) izvršna jedinica nije pokrenuta – nedostaje u ciklusu,
 - b) izvršna jedinica je pokrenuta i traje duže od vremena najgoreg slučaja njenog izvršavanja i
 - c) izvršavanje jedne izvršne jedinice prekinuto od strane druge, što dovodi do njenog produženog trajanja.

Sam alat se sastoji iz dva dijela. Prvi dio rješenja čini modul za učitavanje i obradu mjernih rezultata, te generisanje strukture podataka potrebne za pravilan prikaz. Pod učitavanjem i obradom ulaznih podataka, odnosno rezultata mjerenja performansi sistema, se ne podrazumijeva samo učitavanje informacija iz baze podataka i njihovo filtriranje na osnovu korisničkih zahtjeva, već i dobavljanje dodatnih informacija o entitetima sistema iz različitih fajlova određenih modelom sistema, kako bi vizualizacija imala sve neophodne podatke.

Drugi dio alata se odnosi na samu vizualizaciju podataka. Ovaj dio rješenja obezbjeđuje vizuelni prikaz generisanih podataka, na način koji omogućava njihovo lako

razumijevanje i analizu. Alat donosi mehanizme za lako manipulisanje vizuelno prikazanim podacima, čime je ostvarena i podrška za interakciju korisnika sa samim grafičkim prikazom. Dodatno, drugi dio koncepta rješenja nudi mogućnost proširenog prikaza entiteta sistema koji su korisniku od interesa, čime je znatno olakšan manuelni rad inženjera izdvajanjem i naglašavanjem problematičnih slučajeva izvršavanja softvera.

Na slici 3 prikazan je tok upotrebe alata za vizualizaciju *AUTOSAR logging/tracing* protokola, implementiranog u ovom radu. Kao prvi korak u procesu generisanja prikaza izvršavanja softvera u *AUTOSAR* sistemu, izdvaja se prikupljanje ulaznih podataka na kojima se bazira sam prikaz. Kako se može zaključiti sa slike 3, dio podataka dobija se direktno iz mjerenja izvršavanja softvera, pomoću eksternih alata, dok se drugi dio neophodnih informacija dobavlja iz različitih konfiguracionih fajlova unutar samog sistema. Prikupljeni podaci predstavljaju ulaz u modul za analizu i obradu informacija, na osnovu čega se generiše struktura podataka pogodna za samu vizualizaciju. Data struktura podataka sadrži sve relevantne podatke koji se prosljeđuju dijelu rješenja za grafički prikaz, što ujedno predstavlja i izlaz alata za vizualizaciju *AUTOSAR logging/tracing* protokola.



Slika 3. Koncept rješenja implementacije alata

Na osnovu datog opisa koncepta rješenja, urađena je implementacija programskog rješenja, o čemu će više detalja biti u nastavku rada.

3. REALIZACIJA

Za prikupljanje i obradu podataka neophodnih za kreiranje vizualizacije *AUTOSAR logging/tracing* protokola, korišćen je programski jezik *Python*, koji svojom jednostavnošću upotrebe i bogatstvom biblioteka različitih namjena, nudi veoma lak i efikasan način za implementaciju različitih softverskih rješenja. Za razliku od toga, sam grafički prikaz izvršavanja izvršnih jedinica u automobilskom softveru implementiran je *html + css* tehnologijom, uz dodatak *JavaScript* koncepta.

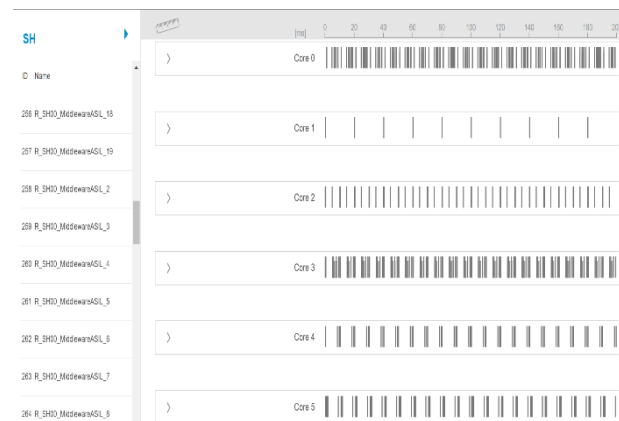
- **Učitavanje podataka dobijenih mjerenjem**

Osnovni preduslov, da bi uopšte izvršavanje softvera u sistemu moglo da se vizualizuje pomoću alata opisanog ovim radom, jeste prikupljanje ulaznih podataka, odnosno mjerenja. Za potrebe rada, korišćen je eksterni alat, čiji

izlaz čini više dokumenata u *.csv* formatu. Najznačajniji od njih, koji predstavljaju ujedno i ulazne podatke za alat za vizualizaciju, jesu *trace_events.csv* i *host_runnable_summary.csv*, u kojima su sadržane sve informacije o realnom izvršavanju izvršnih jedinica sistema, poput vremena u kom se desio neki događaj u sistemu, identifikatora datog događaja ili njegovog tipa. Za obradu datih dokumenata koristi se modul, nazvan *csv_file_handler.py*, u kom je implementirana kompletna logika za učitavanje i obradu informacija o realnim događajima u sistemu. Završni rezultat obrade datih podataka predstavlja kolekcija objekata klase *Slot* koji predstavljaju izvršne jedinice u sistemu. Navedena kolekcija se dalje prosljeđuje kao ulaz glavnom modulu alata, u kom je implementirano glavno rješenje za pripremu podataka da bi vizuelni prikaz bio generisan sa kompletnim skupom detalja o izvršavanju svakog entiteta.

- **Modul za obradu učitanih podataka i generisanje strukture za vizuelni prikaz**

Osnovni algoritam za analizu učitanih ulaznih podataka, te otkrivanje potencijalno nestabilnih ciklusa izvršavanja softvera, implementiran je u okviru *hyperperiod_calculation.py* modula. Osim navedenog, dati modul posjeduje logiku za dobavljanje dodatnih informacija o izvršnim jedinicama, koje nisu dio ulaznih podataka, a služe za generisanje potpunog prikaza. Sam algoritam za obradu rezultata mjerenja performansi, funkcioniše na način da se za svaki ciklus izvršavanja softvera provjeravaju njegova potencijalna odstupanja od idealnog rasporeda izvršavanja, definisanog arhitekturom sistema. Ukoliko se detektuje bilo kakvo odstupanje, dati ciklus i potencijalno problematična izvršna jedinica unutar njega se označavaju, na način da bivaju lako uočljive u vizuelnom prikazu.



Slika 4. Vizualizacija jednog ciklusa izvršavanja

Nakon završetka obrade ulaznog skupa podataka, proširenog dodatnim informacijama o svakoj izvršnoj jedinici, formira se baza podataka u formi *.json* dokumenta, odakle će informacije biti čitane od strane dijela alata za generisanje vizuelnog prikaza. Baza podataka, zapravo, sadrži listu ciklusa izvršavanja, gdje svaki ciklus sadrži realne podatke o svakoj izvršnoj jedinici, o trenutku početka i kraja njenog izvršavanja, te da li je svojim izvršavanjem narušila globalni vremenski raspored. Nakon upisa podataka u bazu podataka, automatski se pokreće procedura njihovog učitavanja i

generisanja prikaza, pri čemu automatski dolazi do otvaranja internet pretraživača i pojavljivanja generisanog prikaza u njemu, što se može vidjeti na slici 4.

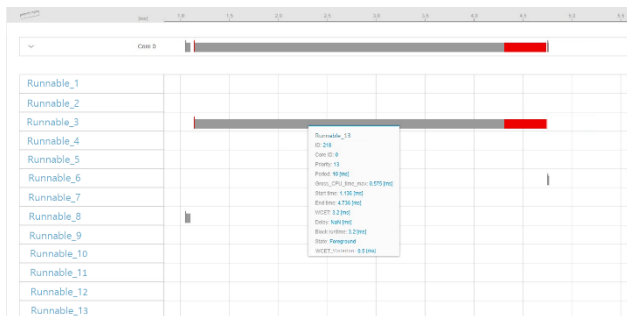
Uspješnim generisanjem grafičkog prikaza događaja, postignut je glavni cilj rada. Sve akcije koje je dalje potrebno preduzeti jesu na inženjeru koji vrši analizu ponašanja sistema, a donošenje odluka je značajno uprošteno i olakšano uz posjedovanje ovakvog alata.

4. TESTIRANJE I REZULTATI

U sklopu testiranja programskog rješenja, u obzir su uzeti slučajevi, odnosno zahtjevi navedeni u poglavlju Koncept rješenja. Svaki od testova urađen je nad različitim ulaznim podacima, kao i različitim podešavanjima sistema, kako bi što veći broj kritičnih slučajeva primjene alata bio pokriven.

Prvi i osnovni slučaj koji je testiran jeste samo učitavanje mjernih podataka iz stabilnog sistem i generisanje prikaza. Opisana situacija je prikazana na slici 4, gdje se može vidjeti da je alat uspješno izgenerisao prikaz jednog ciklusa izvršavanja izvršnih jedinica, čime je uspješno prošao dati test.

Drugi slučaj koji je testiran jeste slučaj koji se često sreće u praksi, pogotovo u procesu razvoja automobilske softvera, a to je da je izvršavanje određene izvršne jedinice produženo, čime je narušeno ograničenje njenog maksimalnog dozvoljenog vremena izvršavanja u najgorem slučaju. Alat je uspješno prepoznao dati problematični ciklus i izvršnu jedinicu koja narušava stabilnost sistema, što se može vidjeti na slici 5. Dodatno je testiran i prikaz detalja o samom kritičnom događaju, a sa slike 5 se može vidjeti da klikom na datu izvršnu jedinicu korisnik dobija detaljan pregled podataka, čime je alat prošao i ovaj test.



Slika 5. Narušavanje Worst Case Execution Time ograničenja

Treći bitan test kom je podvrgnut alat za generisanje vizualnog prikaza AUTOSAR logging/tracing protokola je sličan prethodnom slučaju, s tim da izvršna jedinica produžava svoje izvršavanje zbog prekida od strane druge jedinice većeg prioriteta, što dovodi do nestabilnosti softvera. Alat je uspješno detektovao i obilježio i ovaj slučaj, čime je potvrđeno da su uspješno implementirani svi zahtjevi navedeni u poglavlju Koncept rješenja.

5. ZAKLJUČAK

Istraživanja sprovedena u toku izrade rada pokazala su da kvalitetno dizajnirani alati, poput rješenja obrađenog u

ovom radu, igraju ključnu ulogu u unapređenju kvaliteta proizvodnje, na način da olakšavaju analize kompleksnih podataka koji proizilaze iz sistema zasnovanih na AUTOSAR standardu. Dokazano je da adekvatna vizualizacija podataka omogućava dublje razumijevanje performansi i ponašanja softverskih sistema u automobilu. Osim toga, važno je naglasiti i značaj alata pri detektovanju i otklanjanju grešaka sistema, što direktno doprinosi kvalitetu i pouzdanosti automobila.

Analizom dobijenih rezultata dokazano je da alat za vizualizaciju AUTOSAR logging/tracing protokola predstavlja pouzdano rješenje, koje se uz minimalne izmjene u pogledu konfiguracije, može primjeniti za analizu i poboljšanje performansi različitih komponenti automobilske softvera.

6. LITERATURA

- [1] K. Senthilkumar and R. Ramadoss, „Designing multicore ECU architecture in vehicle networks using AUTOSAR,“ *Third International Conference on Advanced Computing*, Chennai, India, 2011, pp. 270-275).
- [2] Dragan Samardžija, Milena Milošević, *Duboko učenje za autonomna vozila*, 2021, str. 1-2.
- [3] https://ti.tuwien.ac.at/cps/teaching/courses/deterministic-networking/lecture-slides/detnet04_TTEthernet.pdf [pristupljeno februar 2024.].
- [4] Obermaisser R., *Time-Triggered Communication, Networked Embedded Systems*, CRC Press, 2017.
- [5] Navet N., Simonot-Lion F., *Automotive Embedded Systems Handbook*, CRC Press, 2021.
- [6] <https://www.autosar.org/standards/classic-platform> [pristupljeno februar 2024.].
- [7] Qiang Wang, Baiyu Xin, Chao Li and Hong Chen, „The realization of reusability in vehicular software development under AUTOSAR,“ *IEEE Conference Anthology*, China, 2013, pp. 1-6.
- [8] https://www.autosar.org/fileadmin/standards/R22-11/CP/AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf [pristupljeno februar 2024.].
- [9] S. Raju, G. Jeyakumar, A. Mukherji and J. K. Thanki, „Time synchronized diagnostic event data recording based on AUTOSAR,“ *IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, Bhubaneswar, India, 2017, pp 1-6.
- [10] M. Rotim, „Adaptive AUTOSAR Diagnostic Log and Trace processing platform,“ *45th Jubilee International Convention on Information, Communication and Electronic Technology (MIPRO)*, Opatija, Croatia, 2022, pp. 101-105.

Kratka biografija:



Mladen Planojević rođen je na Sokocu 1998. god. Master rad na Fakultetu tehničkih nauka iz oblasti Elektrotehnike i računarstva – Automobilske softver odbranio je 2024.god. kontakt: mladenplanojevic@gmail.com