Računalna glazba

Gugo, Duje

FILOZOFSKI FAKLUTET, SVEUČILIŠTE U ZAGREBU

ODSJEK ZA INFORMACIJSKE I KOMUNIKACIJSKE ZNANOSTI

AK. GODINA 2018./2019.

Duje Gugo

RAČUNALNA GLAZBA

završni rad

Mentor: dr. sc. Tomislav Ivanjko

Zagreb, 2019.

Izjava o akademskoj čestitosti

Izjavljujem i svojim potpisom potvrđujem da je ovaj rad rezultat mog vlastitog rada koji se temelji na istraživanjima te objavljenoj i citiranoj literaturi. Izjavljujem da nijedan dio rada nije napisan na nedozvoljen način, odnosno da je prepisan iz necitiranog rada, te da nijedan dio rada ne krši bilo čija autorska prava. Također izjavljujem da nijedan dio rada nije korišten za bilo koji drugi rad u bilo kojoj drugoj visokoškolskoj, znanstvenoj ili obrazovnoj ustanovi.

_____

(potpis)

Kazalo

Uvod 4
1. Povijest i začeci računalne glazbe 5
2. Stvaranje računalne glazbe 9
2.1. Digitalni zvuk i njegove mogućnosti 10
2.2. Digital Audio Worstations 14
3. Programiranje u skladanju 15
3.1. MUSIC-N jezici za računalnu glazbu 16
3.2. Programski jezici u stvaranju glazbenih računalnih sistema 18
3.3. Moderni programski jezici za računalnu glazbu 19
4. Algoritamsko skladanje glazbe 20
5. Platforme za računalnu glazbu 22
5.1. Desktop platforma 22
5.2. Mobilna platforma 23
5.3. Web preglednici 25
Zaključak 27
Literatura 28
Sažetak 29

Uvod

Računalna glazba je način stvaranja glazbe gdje računalo ima glavnu ulogu u kompoziciji, izvedbi ili soničnoj realizaciji. Računalo u takvom tipu skladanja može služiti kao instrument sa kojim čovjek sklada vlastitu glazbu ili može nezavisno stvarati skladbe putem računalnih algoritama, koju onda može izvoditi čovjek ili samo računalo. To je u srži spoj računalne znanosti i muzičkog stvaralaštva, gdje današnji moderni računalni sustavi uvelike proširuju granice kod mogućnosti stvaranja glazbe. Računalo može sintetizirati bilo koji zvuk kojeg zvučnik može reproducirati te su programski jezici radi svojih širokih mogućnosti primjene savršen alat za upotrijebiti pri skladanju, stoga nije iznenađujuće da su glazbenici prvi umjetnici koji su znatno iskoristili potencijal računalne tehnologije.[1]

U ovom radu govorit će se o počecima računalne glazbe, o prijelazu s analognog zabilježavanja zvuka na digitalno pa sve do modernijih oblika računalne glazbe, o njihovom značaju i prednostima u stvaralaškom svijetu glazbe te o različitim metodama njenog stvaranja. Objasnit će se proces stvaranja računalne glazbe kroz spektar digitalne obrade zvuka te navesti mogućnosti digitalnog zvuka. Uz to će se objasniti uloga programiranja u skladanju i navesti kako su implementirani programski jezici u proces skladanja te će se opisati polje algoritamskog skladanja. Za kraj će se navesti sve raspoložive platforme za stvaranje računalne glazbe. Bazna literatura za pisanje ovog rada je „Computer Music Tutorial“ od Curtis Roadsa koja pruža tehničku i znanstvenu podlogu te „The Oxford Handbook of Computer Music“ od Roger T. Deana koja daje više fenomenološki prikaz računalne glazbe.

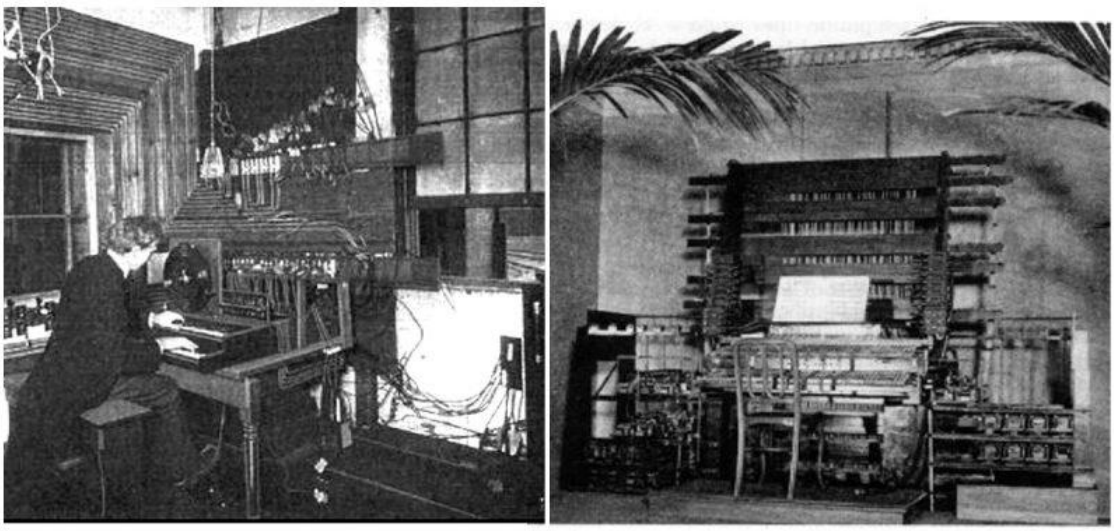
[1] Roads, C. (1996) The Computer Music Tutorial, Predgovor.

1. **Povijest i začeci računalne glazbe**

Sa upotrebom računala i digitalnih uređaja, procesi glazbene kompozicije i njene produkcije su isprepleteni sa znanstvenim i tehničkim resursima društva više nego ikad prije, međutim, upotreba tehnologije u glazbi nije nova stvar.[2]

Thaddeus Cahill je 1906. u Holyoke Massachusetts prvi put prezentirao javnosti svoj Telharmonium, uređaj za stvaranje zvuka koristeći električnu energiju kojeg je patentirao 1897. godine. Sami uređaj je zapravo bio modificirani električni dinamo, koji je sadržavao brojne osovine i pripadajuće induktore kako bi proizveo izmjenične struje različitih audio frekvencija. Ovi signali su prolazili kroz polifoničnu glazbenu tipkovnicu i pripadajuću banku kontrola do serije telefonskih prijemnika opremljenih sa posebnih akustičnim rogovima. Koristeći se telefonskom mrežom je mogao odašiljati zvuk od mjesta gdje se nalazio do hotela, restorana i domova gdje bi se zvuk mehanički amplificirao rogovima. Bio je težak 200 tona, dugačak oko 18 metara i koštao je oko 200,000 dolara, no bio je uređaj koji je nudio mogućnosti produkcije zvuka koje su bile posve nove i fleksibilne do razine koja se nije dosegla dugo nakon njegovog izuma. Unatoč toj činjenici, komercijalno nije stekao popularnost jer je svojim djelovanjem znatno ometao telefonske linije pa je nakon izbijanja prvog svjetskog rata pao u zaborav. Osim zamjene za konvencionalne instrumente sa tipkama je bio i moćan alat za istraživanje širokog spektra električno sintetiziranih tonova. To je bio prvi i najveći zvučni sintetizator ikad napravljen.[3]

[2] Roads, C. (1996) The Computer Music Tutorial. The MIT Press, Predgovor.
[3] Manning, P. (2004) Electronic and Computer Music, Oxford University Press, str. 4



Slika 1 Telharmonium konzola, 1897 [4]

Snimanje glazbe, u smislu kreiranja zvučnih zapisa, ima bogatu povijest koja je započela sa Thomas Edisonovim izumom fonografa 1877. godine, prvog uređaja koji je bio sposoban snimati i reproducirati zvuk. Magnetski žičani snimači su izumljeni 1898. pa nakon njih magnetofonski snimači 1935. u Njemačkoj.[5] Svi ovi uređaji su se koristili u skladanju glazbe prije 1950-ih godina. Tehnologija snimanja zvuka je u ranim 40-ima rodila novi oblik experimentalne glazbe nastale u Parizu zvane *Musique concrète,* gdje bi se koristili isječci snimljenih prirodno proizvedenih zvukova u stvaranju nove glazbe. Mogao bi se nazvati pretečom današnjem sempliranju zvukova kojeg ćemo spomenuti kasnije u radu. Direktna antiteza tom pokretu je bio pokret iz Kölna u Njemačkoj zvan *Elektronische musik*, gdje se zagovaralo korištenje samo elektronički sintetiziranih zvukova pri skladanju.[6]

Analogni snimači poput magnetofona imaju svoje fizičke nedostatke pri zvučnom zapisu. Kada se stvori kopija analognog zapisa istim procesom snimanja, ona će uvijek biti lošije kvalitete jer taj način snimanja dodaje šum u zapis, dok digitalnim načinom snimanja će kvaliteta kopije biti iste kvalitete kao original.[7]

Modularni analogni sintisajzeri 60ih i 70ih koji su stvarali zvuk na principu kontrole voltaže (engl. *voltage control)* te preko glazbenih tipkovnica „čitali" gesture glazbenika putem „okidača" (pulsevi koji se odašilju kada korisnik pritisne tipku) su u to vrijeme uvelike

[4] Georgievska, M. (2017) The Telharmonium is considered to be the first significant electronic musical instrument. URL: https://www.thevintagenews.com/2017/01/05/the-telharmonium-is-considered-to-be-the-first-significant-electronic-musical-instrument/ [17.09.2019.]
[5] Roads, C. (1996) The Computer Music Tutorial. The MIT Press, str. 7
[6] Dean, R.T. (2011) The Oxford Handbook of Computer Music. Oxford University Press, str.17
[7] Roads, C. (1996) The Computer Music Tutorial. The MIT Press, str. 21

proširili paletu mogućnosti sintetiziranja zvukova, no nisu imali tu programabilnost, preciznost, inteligenciju te sposobnost pohrane koju računala imaju.[8] Tek sa uvođenjem pretvarača analognog signala u digitalni (engl. *ADC - analog to digital converter*) te digitalnog u analogni (engl. *DAC - digital to analog converter*) se počela stvarati glazba koja se dotad najviše mogla smatrati računalnom. Sami proces pretvaranja signala će se objasniti kasnije u radu.

Sa uvođenjem pretvarača digitalnog signala u analogni je 1957. Max Mathews u Bell Telephone laboratorijima (engl. *Bell Telephone Laboratories*) počeo istraživati i eksperimentirati sa digitalnom sintezom zvuka putem računala te je iste godine napravio MUSIC I, program sa kojim je proizveo prve zvukove sintetizirane digitalnim računalom. Ubrzo je Matthews došao do realizacije da računala u principu mogu stvoriti bilo koji zvuk. Naime, zbog ondašnjih sporijih procesora bi vrijeme trajanja procesa sintetiziranja bilo znatno veće od trajanja samog proizvedenog zvuka. Drugim riječima, proces proizvodnje zvuka nije bio trenutačan (engl. *real-time*) te bi znanstvenici morali čekati čak po par sati da bi mogli čuti krajnji proizvod svog uloženog rada.[9]

Osim toga su Mathews i njegov tim morali putovati iz svog labosa u New Jerseyu do IBM-ovog glavnog sjedišta jer je jedino njihovo IBM704 računalo bilo dovoljno jako da provede izračune sinteze zvuka. Nakon toga bi se sa digitalnom magnetskom trakom vratili u Bell laboratorij gdje bi na svom računalu slabije moći procesuiranja sa konverterom digitalnog signala u zvuk (engl. *digital to sound converter*) pretvorili uzorke na traci u zvuk. Taj konverter je u to vrijeme bio jedini na svijetu koji je imao mogućnost proizvodnje zvuka. MUSIC V, razvijen 1968. godine, je bila kulminacija Mathewsovog usavršavanja računalne sinteze zvuka te je skoro cijeli bio napisan u Fortranu IV, koji je standardni programski jezik. Eksportirao se mnogim sveučilištima i laboratorijima u ranim 70-ima i mnogim glazbenicima je bio prvi pogled u umijeće digitalne sinteze zvuka. Međutim, radi nedovoljno brzih procesora glazba se i dalje nije mogla sintetizirati trenutačno (engl. *real-time)*.[10]

Definitivni pionir računalne glazbe je Lejaren Hiller, doktorirani industrijski kemičar, koji je prvi koristio računala u procesu algoritmičnog skladanja zajedno sa kolegom

[8] Roads, C. (1996) The Computer Music Tutorial. The MIT Press, str. 623
[9] Dean, R.T. (2011) The Oxford Handbook of Computer Music. Oxford University Press, str. 20-22
[10] Roads, C. (1996) The Computer Music Tutorial. The MIT Press, str. 87-90

Isaacsonom 1956. godine, gdje su na Illiac računalu u Sveučilištu u Illinoisu (engl. *University of Illinois)* kodirali u „sirovom" binarnom strojnom jeziku te stvorili prvu skladbu komponiranu računalom: „*The Illiac suite for string quartet".* Osim toga je i u suradnji sa Robertom Bakerom razvio prvi programski jezik specifične namjene za skladanje zvan MUSICOMP 1963. godine.[11]

Znanstvenik John Chowning sa Sveučilišta Standford je u 1960-ima uz veliku pomoć sintetske moći računala počeo istraživati karakteristike zvukova dobivenih modulacijom frekvencije te je razvio efektivniju metodu sinteze zvuka od aditivne i subtraktivne sinteze (koje su zvuk stvarali tehnikama fiksiranog spektra zvuka), zvanu FM sinteza (engl. *frequency modulation synthesis*) 1973 godine. Uspio je razviti način sinteze „umjetnog" zvuka koji je imao karakteristiku animiranog spektra prirodnih zvukova. Razvio je patent na metodu 1975. u suradnji sa firmom za glazbala Yamaha (tada zvanom Nippon Gakki). FM sinteza je pokrenula masivnu glazbenu revoluciju u muzičkoj industriji jer je daljnje proširila paletu mogućnosti stvaranja zvukova.[12]

Zahvaljujući proboju u dizajnu silikonskih pločica su sredinom 70-ih postali dostupni *real-time* digitalni sintisajzeri (kao zasebni muzički instrumenti), posebno prilagođeni uređaji predviđeni isključivo za stvaranje sintetiziranog zvuka. Naime, u to vrijeme su bili iznimno skupi. Yamahin GS1 digitalni sintisajzer koji je koristio Chowningovu tehniku FM sinteze je koštao 16,000 dolara. Tek sa izlaskom Yamahe DX7 1983. godine je digitalna sinteza u obliku muzičkog instrumenta postala dostupna široj populaciji glazbenika te im otvorila vrata u svijet digitalnog zvuka.[13]

MIDI protokol (engl. *musical instrument digital interface*) je hardverska i softverska specifikacija za prijenos kontrolne informacije između instrumenata i računala te je uveden krajem 1982. kao tehnički standard. MIDI protokol je ključna inovacija jer je bio dizajniran upravo za *real-time* upravljanje glazbenih uređanja, drugim rječima, trenutačnu sintezu zvuka.[14] Više o njemu će se raspraviti kasnije u radu.

---

[11] Roads, C. (1996) The Computer Music Tutorial. The MIT Press, str. 830

[12] Manning, P. (2004) Electronic and Computer Music, Oxford University Press, str. 193-195
[13] Manning, P. (2004) Electronic and Computer Music, Oxford University Press, str. 222
[14] Swift, A. (May 1997) "A brief Introduction to MIDI", Imperial College of Science Technology and Medicine. Dostupno na: https://www.doc.ic.ac.uk/~nd/surprise_97/journal/vol1/aps2/ [17.09.2018]

Što se tiče digitalnog snimanja zvuka, prvi digitalni snimač zvuka je bio Sonyev PCM-1 procesor iz 1977. koji se snimao na videokazete Beta formata, no nije bio dostupan široj populaciji ljudi. Digitalni zvuk je prvi put došao do potrošača 1982. godine u obliku CD formata, optičkog diska koji se čitao laserom, no tek u 80-ima su počeli ugrađivati DAC-ove (engl. *digital to analog converter)* u osobna računala, što je započelo novu eru računalne glazbe. Vrlo brzo nakon toga su sinteza, snimanje i procesuiranje zvuka postali široko rasprostranjeni i dostupni svakom vlasniku PC-a sa dovoljno brzim procesorom. Brži procesori su omogućili da se funkcionalnost sintisajzera implementira u obliku softvera (engl. *DAW - digital audio workstation*) koji u sebi sadrži virtualne instrumente te koji obično nudi grafičku i MIDI kontrolu simulacija starijih akustičnih, analognih i digitalnih instrumenata.[15] Mogućnost preciznog editiranja, manipulacije većim brojem audio traka odjednom te mnoge druge značajke programa su omogućile skladateljima stvaranje i montažu zvuka sa razinom kvalitete i kontrole koje nisu nikako mogli doseći u radu sa analognim trakama. Od sredine do kraja 90-ih su osobna računala imala dovoljno brze procesore da bi Csound, nasljednik MUSIC-N programa i program koji prvotno nije bio sposoban za *real-time* sintezu, mogao u „stvarnom vremenu“ direktno sintetizirati zvuk preko audio izlaza (npr. zvučnika) računala iz datoteka programa ili MIDI ulaznih informacija.[16]

2. Stvaranje računalne glazbe

Makar tradicionalni glazbeni instrumenti imaju vlastiti bogati zvučni prostor i tonalnu specifičnost, prošla su mnoga desetljeća otkad bi skladatelji iz potrebe širenja palete izražavanja u svojoj mašti „stvarali“ zvukove na temelju interpolacije i ekstrapolacije zvukova koje čujemo u prirodi, no koji ne bi bili ostvarivi sa akustičnim ili analognim elektroničkim instrumentima. Generalnost računalne sinteze implicira iznimno veći zvučni prostor u stvaranju, drugim riječima, ona je most između onog što se može zamisliti i onog što se može čuti. Velika zapreka skladateljima je bio nedostatak znanja koje im je potrebno da efektivno izdaje naredbe računalu u procesu sinteze. Manjim djelom je to bilo tehničko znanje u programiranju računala, a većim djelom znanje samog fizičkog i perceptualnog svojstva zvuka. Međutim, fizika, psihologija, matematika te računalna znanosti su sa svojim konceptima u integraciji sa glazbenom naukom i čovjekovom slušnom osjetljivošću se dokazali efektivnima te omogućili glazbenicima, znanstvenicima i tehničarima da u

[15] Dean, R.T. (2011) The Oxford Handbook of Computer Music. Oxford University Press, str. 28
[16] Dean, R.T. (2011) The Oxford Handbook of Computer Music. Oxford University Press, str. 54

zajedničkom radu stvore nove koncepte te fizičke i psihofizičke opise zvuka dovoljne detaljne da postaju korisne skladatelju u realizaciji zvukova iz njegove mašte.

Zato sada postoji dublje shvaćanje boje zvuka te skladatelji imaju bogatiju zvučnu paletu sa kojom raspolažu; razvijene su nove učinkovite tehnike sinteze koje se temelje na modeliranju perceptualnih karakteristika zvuka više nego fizičkih; osim toga su razvijeni i moćni programi za montažu i miksanje sintetiziranog i digitalno nasnimljenog zvuka te su dizajnirani i izgrađeni računalni sintisajzeri posebne namjene. Ovi sistemi za *real-time* izvedbu glazbe inkorporiraju mnoge napretke u znanju i tehnici.[17]

2.1. Digitalni zvuk i njegove mogućnosti

Procesuiranje digitalnog signala u računalnom softveru je omogućilo razvoj mnogih novih tehnika za pretvorbu zvuka te je zamutilo granice između prirodnog i sintetiziranog zvuka. Sa povećanjem opsega računalne memorije se razvila tehnika sinteze sampliranjem, gdje bi se trajanje snimljene note moglo nasnimiti za bolji osjećaj realizma.[18]

Sampliranje ili uzorkovanje je takoreći most između analognog i digitalnog signala. Pojednostavljenim riječima je to proces konvertiranja signala u numeričku vrijednost.[19] Signal sam po sebi je oblik informacije koja putuje nekim fizičkim medijem, u slučaju glazbe je to zrak, jer zvuk nastaje vibracijom čestica zraka koja se onda može prenijet na sljedeći medij poput bubnjića u ljudskom uhu ili mikrofona.[20]

Kada se snima zvuk mikrofonom, mikrofon transducira zvuk kojeg prima u analogni signal, odnosno električne napone koje putuju žicom do ADC-a (engl. *analog to digital converter)* koji pretvori napone u linije binarnih brojeva (uzorkovanje). Binarni brojevi koji definiraju karakteristike uzorka se zatim pohranjuju u unutarnju memoriju, bilo to u obliku tvrdog diska ili nekog eksternog medija za pohranu. Pri reprodukciji zvuka se učitavaju binarni brojevi iz memorije jedan po jedan te prolaze kroz DAC (engl. *digital to analog converter)* koji pretvara linije binarnih brojki u seriju električnih napona. Naponi se filtriraju u

[17] Roads, C. (1996) The Computer Music Tutorial, The MIT Press, Predgovor, Programming and Composition

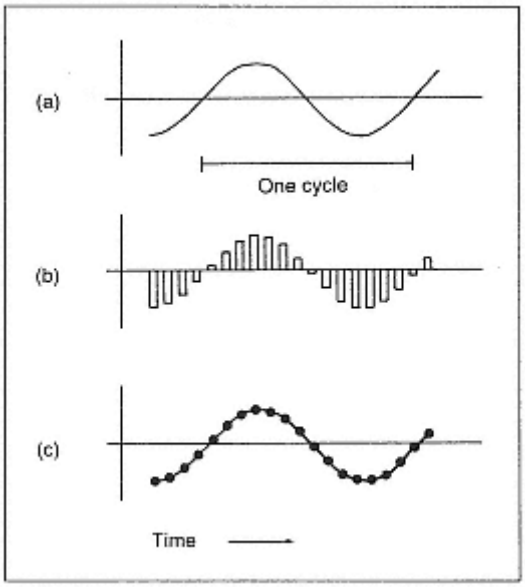
[18] Dean, R.T. (2011) The Oxford Handbook of Computer Music. Oxford University Press, str. 21

[19] Shannon, Claude E. (1949) Communication in the presence of noise, str. 10

[20] Lazzarini, V. (2018) Computer Music Instruments Foundations, Design, and Development. Springer-Verlag, str. 4

valni oblik koji se amplificira te šalje u zvučnik. Vibracije njegove membrane stvaraju zvuk koji čujemo kad slušamo audio zapis sa nekog digitalnog medija.[21]

Radi bolje predodžbe samog procesa je u slici 2 priložena vizualna reprezentacija analognog i digitalnog signala. Sekcija (a) prikazuje analognu valnu sinusoidu, dok sekcija (b) sempliranu verziju sinusoide iz (a), kakva može izgledati na izlazu iz ADC-a. Svaki vertikalni stupac predstavlja jedan uzorak (engl. *sample*). Svaki uzorak je pohranjen u memoriju kao broj koji predstavlja visinu stupca. Sekcija (c) prikazuje rekonstrukciju semplirane verzije sinusoide iz (b) koja nakon procesa filtracije tvori valnu duljinu zvuka koju naposlijetku slušatelj može čuti.



Slika 2 Analogna i digitalna reprezentacija signala [22]

Računalo ne samo da nudi neograničenu teksturalnu kompleksnost u zvuku te stvaranje kompliciranih ali preciznih ritmova, nego i kontrolu i rezoluciju kod parametara poput visine tona, boje zvuka te prostorne lokacije zvuka. Specificiranjem frekvencije se može proizvesti ton bilo koje visine, preciznog i stabilnog štima, tako da nekad izazovne stvari poput stvaranja melodija napisanih u egzotičnim mikrotonalnim ljestvicama postaju iznimno lake.[23]

[21] Roads, C. (1996) The Computer Music Tutorial. The MIT Press, str. 22
[22] Izvor slike: Roads, C. (1996) The Computer Music Tutorial. The MIT Press, str. 25
[23] Dean, R.T. (2011) The Oxford Handbook of Computer Music. Oxford University Press, str. 22

Shvaćanje i manipuliranje zvuka na mikroskopskoj razini, u smislu razumijevanja fizičkih karakteristika valnog oblika zvuka i načina na koje se može utjecati na njega, ujedno i obogaćuje vokabular skladatelja. Digitalna sinteza glazbeniku ne omogućava samo da sklada sa zvukom, omogućava i da „sklada“ sami zvuk. Ne samo to, nego i omogućava potpunu kontrolu nad prostornom slikom zvuka, odnosno stvaranje virtualnog prostora u zvuku simulacijom reverberacije, odzvanjanja zvuka. Tako, recimo, se može izmanipulirati zvuk da zvuči kao da je snimljen u hodniku, crkvi ili pak u špilji i time se daje dodatan ambijentalan prizvuk glazbi.[24]

[24] Roads, C. (1996) The Computer Music Tutorial. The MIT Press, str. 451

2.1.1. MIDI – Musical Instrument Digital Interface

MIDI protokol kojeg smo spomenuli ranije u radu je u principu međusklop povezanosti instrumenata i računala, odnosno set smjernica za prijenos podataka od jednog instrumenta do drugog te uz to i jezik za prijenos informacije o glazbenoj notaciji između računala i sintisajzera koji je primarno dizajniran za *real-time* kontrolu muzičkih uređaja. Osim što utvrđuje shemu međusobne povezanosti hardvera te metodu komunikacije podacima on određuje gramatiku enkodiranja informacije o procesu izvođenja glazbe (kontrolna informacija).

MIDI informacija je pohranjena u obliku malih poruka koje se šalju od jednog uređaja do drugog. Naprimjer, poruka može odrediti početak izvršavanja muzičke note i kraj, visinu njenog tona te njenu početnu amplitudu. Poruke se mogu usporediti sa starinskim rolama za pianolu (automatizirani piano), koje su isto bile skup kontrolnih infomacija u obliku perforacija i rupica na roli papira na temelju kojih bi pianola, uz pomoć unutarnjeg mehanizma, stiskala određene tipke ovisno o informaciji koju vuče iz rola.

Sekvenca MIDI nota definira melodiju, dok drugi parametri zvuka su preneseni preko zasebnih tipova poruka. Iako većina MIDI aplikacija šalju samo kontrolnu informaciju, također je moguće prenositi samplirane audio valne oblike preko midija u određenim uvjetima. MIDI poruke koje se šalju od uređaja do uređaja se šalju u serijsko-binarnom obliku, to jest kao serija bitova jedan po jedan, slično kao kod pretvorbe analognog signala u digitalni. Prijenos se događa kada god uređaj „odluči“ poslati poruku drugom. Ovo je najčešće prilikom nekog događaja (engl. *event*), na primjer kada se pritisne tipka na MIDI tipkovnici.

Računalo ili glazbeni uređaj koji ima MIDI operabilnost nam nudi mnoštvo mogućnosti pri sviranju:

1. MIDI odvaja uređaj za unos informacije (recimo glazbenu tipkovnicu) od generatora zvuka (sintisajzera ili engl. *sampler*-a). To omogućuje da se sa jednom tipkovnicom može upravljati više uređaja.
2. Odvajanje kontrole od sinteze znači da bilo koji uređaj za unos informacija (upravljač dahom, MIDI bubnjevi, električna gitara) koji podržava MIDI može upravljat sintisajzerom. Ovo je dovelo do mnoštva inovacija u dizajnu takvih uređaja.

3. Softveri za interaktivnu izvedbu glazbe, algoritmično skladanje, uređivanje notnog zapisa, se mogu izvoditi na računalu da se pritom njegovi rezultati prenose u sintisajzer. Isto tako u suprotnome smjeru se notni zapisi, sekvence, zapisi izvedbe i uzorci (engl. *sample*s) mogu stvarati pa prenijeti na računalo u svrhu uređivanja ili pohrane.
4. MIDI olakšava razvoj glazbenih softvera, bilo „generičnih" (koji ne ovise o tipu uređaja za unos) ili „ciljanih" (softver za specifičan uređaj).
5. MIDI kodove mogu interpretirati i drugi uređaju osim sintisajzera, poput sistema za rasvjetu (engl. *stage lighting*) ili audio miksera zvuka.
6. Preko MIDI-a se podaci o notnom zapisu, uzorku i sekvenceru (engl. *sequencer*) mogu razmjenjivati između uređaja različitih proizvođača.[25]

Pojava MIDI-a je bez sumnje revolucionizirala glazbenu industriju te u kombinaciji sa pojavom grafičkih korisničkih sučelja iznjedrila brojne MIDI sekvencere, urednike zakrpa (engl. *patches*) za sintezu, urednike zvuka te jednostavnije programe za algoritamsko skladanje.

2.2. Digital Audio Worstations

Kako su krajem 1980ih mikroprocesori u računalima postajali brži i tvrdi diskovi su imali sve veću sposobnost pohrane počeli su se razvijati radne postaje za digitalni audio, odnosno DAW-ovi. Pojednostavljeno, DAW je funkcionalno proširena virtualna konzola za uređivanje zvuka (engl. *audio mixing*) i audioprodukciju. [26] Dolaze u raznim konfiguracijama, bilo to samo u obliku softvera na laptopu, u obliku samostojeće konzole do kompleksne kombinacije raznih vanjskih komponenti kontroliranih centralnim računalom. Jedni od najpoznatijih DAW-ova su Cubase, Logic Pro, Pro Tools i Ableton Live.

Neovisno o konfiguraciji, današnji DAW-ovi imaju središnje sučelje koje korisniku omogućava da manipulira sa višestrukim zvučnim zapisima, odnosno da mijenja samu sliku zvuka kroz proces „miksanja" (engl. *audio mixing*). Koriste se za produkciju i snimanje

[25] Roads, C. (1996) The Computer Music Tutorial. The MIT Press, str. 994-995
[26] Dean, R.T. (2011) The Oxford Handbook of Computer Music. Oxford University Press, str. 28

glazbe, govora, radioemisija, televizije, filmske glazbe, zvučnih efekata i za skoro sve djelatnosti gdje je potreban kvalitetan audio zapis. Iznimno su važan dio glazbene industrije.[27]

„Miksanje“ zvuka je proces dovođenja razina audio kanala do ravonteže koja je auditivno zadovoljavajuća osobi koja miksa. Dok osoba miksa često dodaje razne audio „efekte“ (engl. *audio effects*) koji daju zvuku dodatnu dimenziju i transformiraju karakteristiku audio signala. To se zove procesuiranje audio signala (engl. *audio signal processing*). Reverberacija koju smo ranije spomenuli u radu spada pod ovaj tip manipulacije zvukom.

Danas praktički bilo koji korisnik osobnog računala sa instaliranim DAW softverom, zvučnom karticom i dovoljno brzim procesorom se može baviti muzičkom produkcijom i miksanjem, što uvelike odražava koliko je moć procesuiranja računala utjecala na zanimanje muzičke produkcije.[28] (Dean str. 54)

3. Programiranje u skladanju

Programski jezici zbog svoje generalnosti u namjeni imaju golem opseg moguće praktične aplikacije. Upravo zbog toga nije iznenađujuće da su napisani programi u raznim programskim jezicima u razne svrhe glazbene prirode. Oni koji su se pokazali najkorisnijim te sa kojim su skladatelji stekli najviše iskustva su programi za sintezu i procesuiranje zvuka te programi koji preoblikuju glazbene specifikacije glazbenih djela u fizičke namjenjene programima za sintezu zvuka.

Programiranje je kao vještina iznimno vrijedna skladatelju jer mu daje znanje potrebno da shvati cjelokupno djelovanje sistema te kako se efektivno koristiti njim. Osim toga mu daje određenu razinu neovisnosti pri sintezi zvukova, jer kao u tradicionalnoj orkestraciji, odluke donesene pri sintezi tonova su najčešće jako subjektivne. Iz tog razloga je taj cijeli proces sinteze uvelike poboljšan, u smislu mogućnosti, kada skladatelj može slobodno mijenjati algoritme za sintetiziranje. Programiranje same muzičke strukture je još jedna mogućnost koja se nudi kada skladatelj posjeduje programerske vještine. Procesi skladanja koji se mogu

[27] Kefauver, A. i Patschke, D. (2007) Fundamentals of Digital Audio, New Edition. A-R Editions Inc., str. 133
[28] Dean, R.T. (2011) The Oxford Handbook of Computer Music. Oxford University Press, str. 54

formulirati na precizan način se mogu implementirati u obliku programa i time se, recimo, muzičke strukture koje su bazirane na temelju nekog iterativnog procesa mogu prikladno realizirati programiranjem.

Valja još napomenuti da koncepti programiranja mogu predložiti funkcije djelovanja koje inače ne bi skladatelju pali na pamet izvan konteksta programiranja. Ovo predstavlja veliku važnost u skladanju s obzirom da integracija koncepta programiranja u muzičku maštu skladatelja može proširiti granice same mašte. U tom slučaju programski jezik nije više samo alat sa kojim se može izvršiti neki predodređeni zadatak ili funkcija, nego je isto tako i široka osnova strukture sa kojom mašta može uzajamno djelovati.

Za razliku od slučaja skladanja tradicionalnim instrumentima gdje skladatelj ne može utjecati na akustične kvalitete i karakteristike boje zvuka instrumenta, računalna sinteza dopušta upravljanje mikrostrukturom glazbe. U kontekstu računala mikrostruktura glazbe više nije nužno predodređena, već se može slobodno određivati u mašti skladatelja kao svaki drugi njen aspekt prilikom skladanja.[29]

3.1. MUSIC-N jezici za računalnu glazbu

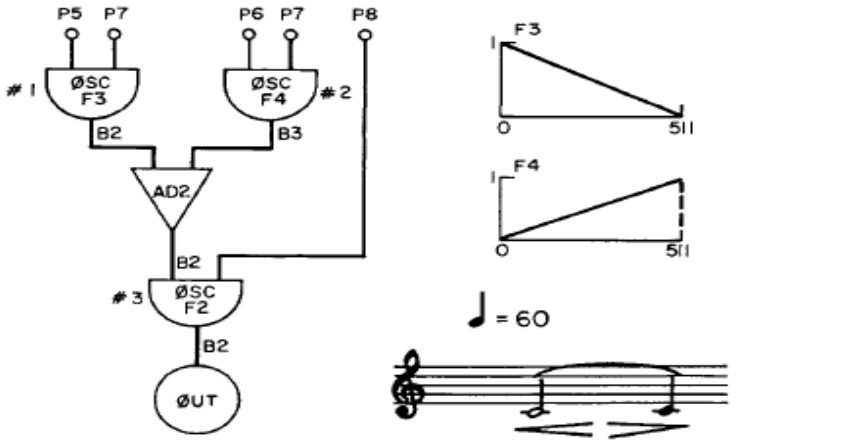
Mathews je osmislio softver za zvučnu sintezu kao dvostruku apstrakciju: orkestar i muzičke notacije. „Orkestar“ je bila kolekcija „instrumenata“ i „instrument“ je zapravo bio set međusobno povezanih „generatora jedinica“, to jest modula za procesuiranje signala. „Notni zapis“ je bio računalni kod koji je sadržavao specifikacije nota koje bi instrumenti svirali. U slici 3 imamo vizualni prikaz samog „instrumenta“. Gore lijevo je dijagram koji prikazuje međusobno povezane generatore jedinica koji stvaraju zvuk. Gore desno su dve funkcije, za diminuendo i krešendo. U sredini desno je sama notacija glazbe koja će se sintetizirati. U donjem djelu slike je MUSIC V „notni zapis“ za sintezu glazbe koji sadrži definiciju instrumenta (linije 1-7) te popis nota (11-12).

U ranijim jezicima za zvučnu sintezu su instrumenti isto bili definirani u notnom zapisu, kasnije bi kod definirao instrumente koji bi bili pohranjeni u jednoj datoteci (datoteka za orkestar) te popis nota u drugoj (datoteka za notni zapis). Mathewsov MUSIC III iz 1960-e je

[29] Roads, C. (1996.) The Computer Music Tutorial. The MIT Press, Predgovor, Programming and Composition.

bio prvi baziran na paradigmi generatora jedinica. Najutjecajniji program je definitivno bio MUSIC V, a najrasprostranjeniji Csound koji se koristi dan danas. [30]

Mnoge implementacije Mathewsa koje su prisutne u njegovom MUSIC-N softveru su norma za većinu hardverskih i softverskih sistema te sistema za procesuiranje digitalnog signala (engl. *DSP software*). MUSIC-N obitelj softvera je iznjedrila mnoge modernije inačice softvera za skladanje poput SuperCollider-a, Max-a, JSyn-a te ChucK-a kojeg ćemo ponovno spomenuti kasnije u radu. [31]




```
 1   INS 0 3 ;
 2   ØSC P5 P7 B2 F3 P30 ;
 3   ØSC P6 P7 B3 F4 P29 ;
 4   AD2 B2 B3 B2 ;
 5   ØSC B2 P8 B2 F2 V1 ;
 6   ØUT B2 B1 ;
 7   END ;
 8   GEN 0 1 3 .999 0 0 511 ;
 9   GEN 0 1 4 0 0 .999 511 ;
10   GEN 0 1 2 0 0 .99 50 .99 205 −.99 306 −.99 461 0 511 ;
11   NØT 0 3 2 0 2000 .0128 6.70 ;
12   NØT 2 3 1 2000 0 .0256 6.70 ;
13   TER 3 ;
```

Slika 3 Jednostavan primjer MUSIC V „instrumenta“[32]

Dio MUSIC-N jezika koji je apstraktirao tradicionalni notni zapis je omogućio skladatelju da popiše note u kompoziciji skupa sa parametrima po noti za instrument koji bi svirao tu notu.

[30] Dean, R.T. (2011) The Oxford Handbook of Computer Music. Oxford University Press, str. 26
[31] Lazzarini, V. (2018) Computer Music Instruments Foundations, Design, and Development. Springer-Verlag, str. 25
[32] Izvor slike: Dean, R.T. (2011) The Oxford Handbook of Computer Music. Oxford University Press, str. 26

Ovisno o jeziku bi različite razine algoritmične kontrole bile dostupne za stvaranje note i definiranje njihovih parametara.

Notni popis MUSIC-N jezika se naime može i očitati ne samo unutar sfere softvera za sintezu zvuka, već se može i prevesti u MIDI podatke za kontrolu hardverskih sintisajzera ili može biti prevedeno u glazbenu notaciju za tradicionalnu „ljudsku“ izvedbu. MUSICOMP kojeg smo ranije spomenuli spada u kategoriju potonjeg primjera.[33]

3.2. Programski jezici u stvaranju glazbenih računalnih sistema

Ideja Max Mathewsa koju je oformirao u eksperimentima sa MUSIC I i MUSIC II programima je da glazbenici, skladatelji i istraživači bi bolje djelovali u sistemu koji ima otvorene mogućnosti, koji je sastavljen od više manjih jedinica koje se mogu povezati međusobno i funkcionirati kao cjelina. Drugim rječima, da se umjesto jednog centraliziranog programskog jezika koristi više njih zaduženih za određene parametre zvuka. Upravo na ovaj način se stvaraju računalni sistemi sastavljeni od više komponenata, u ovom slučaju programskih jezika.

Csound je najpoznatiji primjer programskog jezika koji je nastao iz MUSIC-N obitelji softvera. Csound je kompletan programski jezik za specifične domene (engl. *DSL- domain-specific language)* te okolina za izvedbu koja nudi široke mogućnosti procesuiranja zvuka. On je koristan u dizajnu specifičnih elemenata sistema (npr. elementi vezani uz procesuiranje signala) koje drugi jezici nisu sposobni dizajnirati zbog nedostatka ugrađenih komponenata.

Uz DSL jezike važan element kod glazbenih računalnih sistema su i programski jezici opće namjene (engl. *GPPL- general purpose programming language).* Imaju dve uloge za koje su dobro opremljeni. Prva je da budu vezivo za kontrolne komponente i komponente za procesuiranje signala u našem sistemu. Naprimjer, da omoguće programeru da stvori korisničko sučelje. Najprikladniji jezici za ovaj zadatak su programski jezici više razine (npr. Python), iz razloga što su jako jednostavni za koristiti i nude mnoge mogućnosti za stvaranje sučelja.

[33] Dean, R.T. (2011) The Oxford Handbook of Computer Music. Oxford University Press, str. 26

Druga uloga jezika opće namjene, u ovom slučaju jezika niže razine, je implementacija zadataka koji zahtjevaju intenzivno računanje. Najčešće se koriste jezici implementacijske razine poput C i C++. Određeni elementi sistema koji su zahtjevniji bi trebali biti programirani uz pomoć ovih jezika te kompajlirani (prevedeni u drugi jezik) u komponente koje može učitati DSL jezik. Još jedan važan programski jezik koji se koristi za stvaranje algoritama za procesuiranje digitalnog signala je Faust, koji se može kompajlirati u C++ kod.

Iako se sistem za računalnu glazbu zapravo mogao sastaviti u samo jednom programskom jeziku, svaki od ovih jezika ima svoje specifične računalne prednosti koje nam smanjuju količinu i vrijeme potrebnog programiranja te kad se kombiniraju stvaraju jedan efikasan i pregledan višejezični sistem za stvaranje računalne glazbe koji jamči vrlo bogato računalno-glazbeno iskustvo.[34]

3.3. Moderni programski jezici za računalnu glazbu

Noviji jezici za programiranje glazbe nude veću fleksibilnost. *Nyquist* (Dannenberg 1997) je otklonio separaciju notnog zapisa i orkestra te kontrolnih signala i audio signala. Također nudi korisniku sve mogućnosti funkcijskih programskih jezika (u ovom slučaju LISP). *SuperCollider* (McCartney 2002) je sastavljen od objektno-orientiranog jezika za kompoziciju i servera za *real-time* sintezu kojem klijentske aplikacije mogu pristupiti preko mreže. Postoje brojni drugi softver paketi za sintezu zvuka koji kao bazu koriste funkcijske programske jezike poput *C++* ili *LISP*.

ChucK (Wang i Cook 2003) je noviji tekstualni programski jezik za *real-time* sintezu zvuka. Ima posebnu sintaksu za manipuliranje toka vremena, i njegov temporalni model odstupa od modela N-MUSIC programa. Dizajniran je na način da se fokusira više na čitljivost i fleksibilnost programa za korisnika nego na druge čimbenike poput sirove performanse.[35] Napravljen je da bude koristan za „usputno“ programiranje (engl. *on-the-fly programming),* što se u kontekstu izvedbe zove kodiranje uživo (engl. *live coding*). Kodiranje uživo iskorištava virtualizaciju instrumenta u smislu da izvođač/programer istovremeno

[34] Lazzarini, V. (2018) Computer Music Instruments Foundations, Design, and Development. Springer-Verlag str. 25-26

[35] Ge Wang (2002) ChucK : Strongly-timed, Concurrent, and On-the-fly Music Programming Language, Princeton University Soundlab. Dostupno na http://chuck.cs.princeton.edu/ [17.09.2019]

„gradi“ svoje instrumente dok ih svira. Drugim riječima, *live coder* je improvizirajući programer zvuka.[36]

```
// ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
// BD - bass drum
class BD_808 extends Chubgraph
{
    inlet => BridgedT bt => LPF lp => HardLimit hl => outlet;

    16000 => lp.freq; // tuned by ear

    47.444 => float freq; //
    4.0 => float decay;

    2.0 => bt.gain; // tuned by ear
    bt.setFreq(freq); bt.setDecay(decay);

    fun void noteOn(float accent) // accented when noteOn is passed a float > 0.0
    {
        if (accent != 0.0)
        { // if accent, do trick as described in [1]
            bt.setFreq(freq*2.813; // up an octave
            bt.noteOn();
            4::ms => now;
            bt.setFreq(freq); // fundamental
        } else { noteOn(); } // call unaccented version
    }

    fun void noteOn() // unaccented when noteOn is passed nothing
    {
        bt.setFreq(freq);
        bt.noteOn();
    }
}
```

Slika 4 Primjer jednostavnog ChucK programa koji emulira zvuk bas bubnja[37]

4. Algoritamsko skladanje glazbe

Dok glazbenik sluša glazbu, dio njega upija doživljaj melodija i harmonija u nekoj skladbi, dok drugi dio konstantno predviđa i pretpostavlja kako će skladba teči u smislu dinamike i strukture. Skladatelji su stoljećima znali da se mnogi glazbeni procesi mogu formalizirati u simboličku reprezentaciju. Formalni kompozicijski algoritam je algoritam za kreaciju glazbe, stoga računalo može poslužiti kao stroj za glazbene ideje.

Programi za algoritmičko skladanje se mogu podijeliti na programe koji komponiraju na način da putem algoritama se stvori gotov notni zapis kojeg onda čovjek može čitati pa zatim izvoditi uživo te na programe koji skladaju putem algoritama i pritom sami izvode skladbu. Algoritmične skladbe se mogu podijeliti na: skladbe koje su komponirane uz pomoć računala i na skladbe koje su komponirane od strane samog računala.[38]

[36] Dean, R.T. (2011) The Oxford Handbook of Computer Music. Oxford University Press, str. 27
[37] Izvor slike: https://kurtjameswerner.tumblr.com/post/50274769999/chuck-tr-808-emulator-bass-drum-bd-emulation [17.09.2019.]
[38] Roads, C. (1996) The Computer Music Tutorial. The MIT Press, str. 821

Illiac suite je algoritmična skladba koju je Lejaren Hiller skladao u suradnji sa Isaacsonom 1956. i izdao 1957. te se sastojala od 4 djela, prva dva su bila skladana u tradicionalnom stilu, a druga dva u modernom eksperimentalnom stilu. Tako je ovaj komad računalno skladane glazbe ilustrirao dva tipa algoritmičnog skladanja koja su se otada daljnje razvili: prvi je formalan tip koji emulira neki etablirani tradicionalni stil skladbe te aplicira pravila glazbene teorije (na primjer barok, američki jazz, liturgijska glazba...) kao oblik validacije sposobnosti programa da reproducira neki poznati glazbeni oblik te drugi tip koji je originalan te ovisi o skladateljovoj estetskoj i kreativnoj potrebi. Više skladatelja je skladalo u drugom tipu, pretpostavilo bi se zbog toga što je više skladatelja željelo skladati nešto originalno nego emulirati poznate stilove.[39]

Najpoznatiji u takvom skladanju su bili Hiller i Iannis Xenakis, koji su najviše skladali u stokastičkom principu algoritmičkog skladanja, koje je temeljeno na nasumičnosti parametara u skladbi. Veliku ulogu u ovakvom tipu skladanja igraju Markov lanci koji su, u informacijskoj teoriji, lanci vjerojatnosti u kojima vjerojatnost budućeg događaja ovisi o stanju jednog ili više događaja u prošlosti. U kontesktu glazbe su događaji određene note. Stokastično skladanje je temeljna baza za sva buduće oblike algoritmičkog skladanja.[40]

Postoji mnogo načina skladanja sa algoritmima. Skladatelj može formalnom metodom dizajnirati program sa zadanim određenim parametrima unutar algoritma pa dobivenu skladbu kojeg je skladalo računalo izvesti takvu kakva je. S druge strane formalna algoritmična skladba koju sklada računalno može biti samo početna točka skladanja gdje skladatelj dalje odabire koje dijelove će zadržati i koje izbaciti.[41]

Može skladati sa algoritmom koji će ponuditi većinom jako nasumične odabire nota, instrumenata i ritmova. Isto tako skladatelj može daljnje utjecati na ishod algoritma tako što može postaviti vlastita pravila unutar programa koja na neki način ograničavaju razinu nasumičnosti postavljajući provjeru repeticije određenih djelova u skladbi, kao što je to na primjer radio Rudolf Koenig. On je generalizirao svoju praksu skladanja i izrazio svoje koncepte skladanja kao apstrakcije u softveru. Ovaj tip „suzdržane nasumičnosti“ kojeg smo spomenuli je implementirao u svojim apstrakcijama te je na taj način je stvarao glazbu koja je

[39] Dean, R.T. (2011) The Oxford Handbook of Computer Music. Oxford University Press, str. 33.
[40] Roads, C. (1996) The Computer Music Tutorial. The MIT Press, str. 878.
[41] Roads, C. (1996) The Computer Music Tutorial. The MIT Press, str. 908.

bila iznimno originalna (radi dodatka elementa nasumičnosti) i uz to čvrsto utemeljena u njegovim kompozicijskim teorijama i praksama.[42]

U programima za kompoziciju baziranim na znanju (engl. *knowledge based*), gdje program koristi bazu znanja da rješava zadatke, skladbe se mogu stvarati tako da se izolira „estetski kod" određenog glazbenog žanra te da se kasnije na temelju njega stvore nove skladbe slične toj. Baziraju se na preodređenim setovima argumenata koji se mogu upotrebiti u skladbi novih glazbenih djela istog stila ili žanra.[43]

Još jedan zaista zanimljiv model skladanja je skladanje kroz teoriju prirodnih fenomena, gdje bi se kompozicije stvarale iz harmoničnih i neharmoničnih fenomena u prirodi. Na primjer, od 1970-ih su se fraktali, proučavali kao modeli za algoritmično skladanje.[44]

5. Platforme za računalnu glazbu

Računalne platforme generalne primjene, koje su namjenjene rješavanju šireg spektra problema, nude stabilnu okolinu razvoja i perfomanse za računalnu glazbu. Dopuštaju programima poput MUSIC-V koji se razvio prije 50 godina da se i dalje koristi u modernim sistemima zbog dostupnosti FORTRAN kompajlera, koji mogu uzeti originalan kod programa i uz malu preinaku proizvesti funkcionalan softver. Ova dugotrajna univerzalnost primjene omogućuje da se dalje održe i primjene ideje i koncepti koji su se pojavili tijekom duge povijesti računalne glazbe, pružajući kontinuirani tok razvoja sistema. Danas se primarno koriste računala generalne namjene te programi koji su napisani u programskom jeziku široke i multi-platformne namjene.[45]

5.1. Desktop platforma

Moderna desktop platforma je cjelovita okolina za razvoj, produkciju i izvedbu računalne glazbe. Omogućuje korisnicima da navigiraju kroz široki raspon softverskih okolina,

[42] Dean, R.T. (2011) The Oxford Handbook of Computer Music. Oxford University Press, str. 75.
[43] Brown, S. (1997) Algorithmic Composition and Reductionist Analysis: Can a Machine Compose? Dostupno na: https://www.doc.ic.ac.uk/~nd/surprise_97/journal/vol1/aps2/
[44] Roads, C. (1996) The Computer Music Tutorial. The MIT Press, str. 856.
[45] Lazzarini, V. (2018) Computer Music Instruments Foundations, Design, and Development. Springer-Verlag, str. 295.

da ih kombiniraju i povezuju, da kombiniraju programske jezike, prikazuju podatke, postavljaju kontrole te stvaraju audio visoke kvalitete trenutačno. Razvoj desktop platforme kao okoline za stvaranje računalne glazbe je ovisio o ključnim komponentima:

1. MUSIC-N paradigmi: set etabliranih određenja unutar sistema za glazbeno programiranje poput Csound-a, čija je temeljna ideja da fleksibilna platforma za računalnu glazbu mora biti programabilna.
2. Objektno orientirano programiranje: MUSIC-N paradigma je realizirana kroz principe objektno orientiranog programiranja.
3. Višejezičnost: kombinacija jezika specifične domene (Csound), jezika za skriptiranje i nekad jezika za implementaciju u sistemu (C, C++) nudi jako fleksibilni i cjeloviti skup rješenja za sintezu zvuka i procesuiranje. Omogućuje odijeljivanje većih dizajnerskih problema u manje zasebne komponente.
4. Besplatni softver i softver otvorenog izbora (engl. *open-source*): dostupnost i razmjena izvornog koda je jedan od glavnih pokretača razvoja računalne glazbe.
5. Ekosistem za stvaranje glazbe: softverski ekosistemi su softveri koji funkcioniraju na kooperativan i komplementaran način na istim ili srodnim platformama.

Koncept ekosistema računalne glazbe se poput paradigme višejezičnosti isto zasniva na komplementarnosti različitih djelova softvera, zato je u modernoj desktop platformi normalno započeti rad unutar jednog tipa aplikacije pa integrirati druge ako se javi potreba za druge specifične operacije. Tako recimo možemo koristiti softvere poput urednika zvuka (engl. *sound editor*) s kojim manipuliramo zvučnu datoteku i učitamo ju u neki DAW projekt pa u potrebi za razvojem algoritama za glazbu integriramo glazbeni DSL jezik za generalnu upotrebu. [46]

5.2. Mobilna platforma

U zadnjih desetak godina, mobilne računalne platforme sa povećanjem računalne moći procesora su se pokazale kao korisna i funkcionalna alternativa desktop platformama. Danas su mobilni uređaji sposobni za korištenje u izvedbi i skladanju glazbe te za dizajn zvuka. Koriste se dva operacijska sistema koji podržavaju aplikacije za računalnu glazbu i to su iOS i

[46] Lazzarini, V. (2018) Computer Music Instruments Foundations, Design, and Development. Springer-Verlag, str. 297- 298

Android, među kojim je iOS uvijek uvijek prednjačio jer pruža bolju audio kvalitetu i bržu reprodukciju.

Ekosistem desktop računalne glazbe se daljnje proširuje korištenjem mobilnih aplikacija. Često neka desktop aplikacija ima mobilnu verziju koja replicira neke njene funkcionalnosti ili ju nadopunjava. Mobilni DAW-ovi su primjer ovog. Makar ne mogu rekreirati kompleksno sučelje koje nudi standardni desktop DAW, dopuštaju određenu razinu uređivanja i stvaranja glazbe „u hodu“ koju i može radi kompatibilnosti datoteka poslati na desktop DAW.

Osim toga postoje i programski jezici specifične domene za mobilne platforme te se stvaraju sučelja za programiranje aplikacija poput CsoundObj API baziranog na Csound jeziku.[47]

[47] Lazzarini, V. (2018) Computer Music Instruments Foundations, Design, and Development. Springer-Verlag, str. 299

5.3. Web preglednici

Današnji web preglednici su postali oblik virtualne platforme, koji djeluju unutar raznih operacijskih sistema i hardvera kojih podržavaju. U nekim slučajevima su oni sami postali operacijski sistem (naprimjer ChromeOS i FirefoxOS). Ujedinjuju ih standardi poput HTML 5 (Hypertext Markup Language) te sistemski jezik Javascript (JS) te se aplikacije šalju preko HTTP-a (Hypertext Transfer Protocol) u obliku običnog teksta kojeg interpretira preglednik.

Preglednici značajno proširuju ekosistem računalne glazbe. Pošto ne ovise o softverskim i hardverskim platformama te se mogu prilagoditi njima, nude okolinu gdje se spajaju lokalni i udaljeni resursi. Aplikacije se pružaju iz vanjskih lokacija, no pokreću se na računalu korisnika. Postoji širok raspon softvera koji djeluju na različitim razinama: web DAW-ovi, softverski sintisajzeri, programske ekstenzije (engl. *plug-ins*), upravljači na daljinu itd. Također je ponovno važna prisutnost glazbenih DSL-a.

Mrežne Javascript aplikacije interpretira preglednik, koji prevede kod u računalne radnje koje se trebaju izvesti. Makar ove aplikacije jesu efikasne, radi toga što ne koriste svoju originalno implementiranu bazu koda (u C ili C++) već se koriste Javascriptom kao posrednikom (engl. *Javascript cross-compilation*) te nisu brze i efikasne poput aplikacija koji koriste takvu kodnu bazu (engl. *natively-implemented programs*). No zato se mogu prikazati u obliku sučelja (engl. *frontend*) PnaCl (engl. *Portable Native Clients)* koje je puno efikasnije te pruža izvrsnu performasu.

Platforma preglednika stvara fluidan odnos između lokalnih i udaljenih računalnih sistema te omogućava široku distribuciju audio aplikacija bez zapreka. Koristi se recimo za koncerte i instalacije gdje se mobilni uređaji publike koriste kao komponente u jednom velikom međupovezanom računalnom instrumentu.[48]

Kombinacijom desktop, mobilnih i pregledničkih platformi dobivamo digitalni ekosistem koji je zapravo skup međuoperabilnih uređaja i servisa koji maksimiziraju kontrolu korisnika nad glazbom koju bi stvarao te zahvaljujući internetu povezuju izvođače glazbe sa drugim izvođačima te sa svojom publikom. Zahvaljujući njemu glazbenici mogu kolaborirati

[48] Lazzarini, V. (2018) Computer Music Instruments Foundations, Design, and Development. Springer-Verlag, str. 300-302

sa drugim umreženim glazbenicima u obliku umreženog muzičkog performansa (engl. *networked music performance*). Uz to mogu i produbiti iskustvo nastupa uživo tako da omogućuju publici da sudjeluje u procesu stvaranja glazbe (engl. *Participatory live music pefrormance systems*), bilo u obliku senzora, uz pomoć smart mobilnih uređaja ili opipljivih sučelja.[49]

[49] Turchet, L. (2012) Towards the Internet of Musical Things, KTH Royal Institute of Technology, str. 2

Zaključak

Računalna glazba kao fenomen je produkt integracije računalne tehnologije u glazbenu umjetničku formu. Zahvaljujući tehnološkim inovacijama u radu sa digitalnim zvukom i internetu danas svaki glazbenik koji posjeduje osobno računalo ima mogućnost kreiranja vlastite računalne glazbe. Višedimenzionalnost i fleksibilnost digitalnog zvuka u kombinaciji sa računalnom moći računala omogućavaju glazbenicima da stvaraju jedinstvene zvučne slike dok im programi za rad sa digitalnim zvukom nude pregršt načina da sintetiziraju zvuk kojeg imaju zamišljenog u mašti. Znanje programiranja se pokazalo iznimno korisno u organizaciji, uređivanju i stvaranju glazbe te u proširenju mogućnosti pri skladanju, a generalnost namjene programskih jezika se pokazala veoma korisnom pri dizajnu računalnih glazbenih sistema. Na kraju ovog rada se može zaključiti da je računalna glazba interesantan produkt ljudske želje za širenjem mogućnosti glazbenog izražavanja te da je zahvaljujući modernim platformama i pojavi interneta dosegla neviđene razine kreativnosti u izražavanju koje će samo rasti uz daljnji napredak tehnologije.

Literatura

1. Brown, S. (1997) Algorithmic Composition and Reductionist Analysis: Can a Machine Compose?

2. Dean, R.T. (2011) The Oxford Handbook of Computer Music. Oxford University Press.

3. Ge Wang (2002) ChucK : Strongly-timed, Concurrent, and On-the-fly Music Programming Language, Princeton University Soundlab. Dostupno na http://chuck.cs.princeton.edu/

4. Kefauver, A.P. i Patschke, D. (2007) The Fundamentals of Digital Audio. A-R Editions Inc.

5. Lazzarini, V. (2018) Computer Music Instruments Foundations, Design, and Development. Springer- Verlag.

6. Manning, P. (2004) Electronic and computer music. Oxford University Press.

7. Roads, C. (1996) The Computer Music Tutorial. The MIT Press.

8. Shannon, Claude E. (1949). Communication in the presence of noise. Proceedings of the IEEE. Izdano 1998

9. Turchet, L. (2012) Towards the Internet of Musical Things. KTH Royal Institute of Technology.

10. Swift, A. (May 1997), A brief Introduction to MIDI, Imperial College of Science Technology and Medicine. URL: https://www.doc.ic.ac.uk/~nd/surprise_97/journal/vol1/aps2/ [17.09.2018.].

Računalna glazba

Sažetak

Radi čovjekove želje za inovacijom je uvijek bila prisutna integracija tehnoloških novina u svim poljima, čak i u polju glazbene umjetnosti. Sredinom dvadesetog stoljeća je zahvaljujući pojavi digitalnog signala i programabilnoj moći modernih računala bio kreiran novi oblik stvaranja i manipulacije sintetiziranog zvuka predvođen inženjerom elektrotehnike Max Mathewsom i njegovim istraživanjima u mladom polju računalne glazbe. Zahvaljujući njima glazbenici su mogli dizajnirati sami „atom zvuka“ u glazbi koje stvaraju te naučiti o tada neviđenim mogućnostima njegovog oblikovanja. Uz to se implementacija programskih jezika u proces skladanja pokazala iznimno uspješnom i korisnom radi generalnosti njihove primjene te su programski jezici kroz kontekst programiranja pomogli glazbenicima otkriti nove tehnike pristupa kompoziciji glazbe. Ovo su samo neke od tema koje su opisane u ovome radu, a osim njih su navedeni i opisani brojni programi i pomagala za stvaranje računalne glazbe te platforme na kojima se može stvarati.

Ključne riječi: računalna glazba, digitalni zvuk, programski jezici, skladanje

Computer Music

Summary

Due to the human strive for innovation there has always been an integration of technological novelties in all fields, including the field of music. In the middle of the 20th century a new way of forming and manipulating synthesized sound was created, thanks to the advent of digital signal processing and the programming power of modern computers. This new wave of synthesized sound processing and manipulation was headed by electronic engineer Max Matthews and his research in the still young field of computer music. Thanks to him and his research, musicians could design the bare "atom of sound" in the music they are creating and also learn about the never before seen possibilities of it's formation. Other than that, the implementation of programming languages in the process of composition has proven itself to be quite successful and useful due to the generality of its application. Also, the programming languages themselves have helped musicians discover new techniques of approaching music composition through the context of programming. These are merely some of the subjects that have been adressed in this work, including the subject of the wide range of applications and tools made for creating computer music and the available platforms for its creation.

Key words: computer music, digital sound, programming languages, composition