KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
PROGRAMŲ INŽINERIJOS KATEDRA

Tadas Mažutis

Trimatės grafikos elementų panaudojimas vizualizacijos sistemose

Magistro darbas

Darbo vadovas
prof. E. Bareiša

Kaunas, 2009

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
PROGRAMŲ INŽINERIJOS KATEDRA

Tadas Mažutis

Trimatės grafikos elementų panaudojimas vizualizacijos sistemose

Magistro darbas

Recenzentas
doc. dr. A. Lenkevičius
2009-05

Vadovas
prof. E. Bareiša
2009-05

Atliko
IFM-3/2 gr. stud.
Tadas Mažutis
2009-05-25

Kaunas, 2009

Turinys

Santrauka

Šio darbo tikslas yra suprojektuoti ir realizuoti vizualizacijos sistemą, naudojančią trimačius elementus ir atlikti tyrimą, ar tokia vizualizacijos sistemos koncepcija padeda spręsti tam tikras operatorių problemas. Darbo metu siekiama išsiaiškinti, ar trimačiai elementai gali būti naudingesni nei dvimačiai jų analogai.

Vizualizacijos sistemai realizuoti panaudotas Microsoft DirectX trimatės grafikos kūrimo priemonių rinkinys. Sukurtas realiai veikiančios dvimatės vizualizacijos sistemos trimatis analogas, apimantis tam tikrą realios sistemos dalį. Atliktas tyrimas, kurio tikslas – išsiaiškinti, ar sukurta sistema turi pranašumą prieš savo pirmtką identifikuojant ir lokalizuojant sistemoje įvykusiems įvykiams.

Summary

Using 3D elements in visualization systems

The goal of this work is to develop vizualization system which uses 3D elements. Also to perform research with goal to analyze if such visualization system helps to solve particular problems for operators. During this work will be trying to answer the question if 3D elements can be more helpful than 2D elements.

Microsoft DirectX SDK is used to create visualization system. Only a specific part of running 2D visualization system is created. Research is performed to identify if created system can be more helpful in identifying and localizing system events.

Paveikslėlių sąrašas

Lentelių sąrašas

Įvadas

Vartotojo sąsajos patogumas, intuityvumas ir funkcionalumas yra pagrindiniai aspektai vertinant vartotojo sąsajos kokybę. Tose situacijose, kada vartotojo sąsaja projektuojama tam, kad butų naudojama įvairiausių procesų (tarp jų realaus laiko sistemų) stebėjimui, ypač svarbu duomenis pateikti taip, kad jie būtų kuo išsamesni bei suprasti vienareikšmiškai. Šioje srityje yra pasiekta ganėtinai daug, sukurta daug įvairių SCADA (Supervisory Control and Data Acquisition) paketų, sugebančių apdoroti ir pateikti didelius kiekius duomenų. Kartais, kuomet duomenys yra panašaus tipo, pvz. informacija apie to paties tipo įrenginius (įvaraus dydžio ir galios siurbliai, ventiliatoriai ir pan.) juos pateikti vienareikšmiškai ir paprastai yra ganėtinai sudėtinga. Tokiose situacijose stengiamasi informaciją apie įrenginį pateikti kartu su paties įrenginio atvaizdu. Tačiau jei įrenginys yra pakankamai sudėtingas, dvimatis jo atvaizdavimas nebesugeba pilnavertiškai pateikti intuityvaus situacijos vaizdo, todėl atsiranda poreikis dvimačius objektus pakeisti trimačiais. Vizualizacijos vertei augant, atsiranda poreikis plėsti ne tik jos funkcionalumą, bet ir patį požiūrį į vizualizaciją.

Trimačių objektų naudojimas vizualizacijos sistemose nėra labai paplitęs dėl kelių priežasčių: tokios vizualizacijos sistemos yra sudėtingos kurti; tokios vizualizacijos sistemos tampa sudėtingos konfigūruoti, t.y. sistemų intergatoriams reikia papildomų žinių, gebėjimų ir patirties; trimačiai objektai yra reiklesni kompiuteriniams resursams; galutinis vartotojas turi būti papildomai specialiai paruoštas tokios vizualizacijos sistemos eksploatacijai. Tai ir yra pagrindinės priežastys kodėl trimačiai objektai tarp vizualizacijos sistemų nėra plačiai paplitę.

Vizualizacijos sistemos yra vertinamos daugiausia techniniu aspektu – į vizualinę pusę yra kreipiamas labai nedidelis dėmesys. O to kokybiškai sukurtai konkretaus projekto vizualizacijai kartais labai trūksta. Trimatė grafika padeda ne tik pateikti papildomos ir geriau suprantamos informacijos, bet taip pat suteikia daugiau galimybių padaryti vizualizaciją patrauklią vartotojui.

Vizualizacija ir jos rūšys

Kalbant apie vizualizaciją plačiąja prasme galime ją įsivaizduoti nuo paprasčiausio modelio popieriuje, maketo ar fotografijos iki sudėtingų automatizuoto projektavimo bei braižymo programinės įrangos paketų, reikalaujančių milžiniškų techninės įrangos resursų. Kadangi perteikti savo mintis vizualiai yra pats suprantamiausias ir labiausiai priimtinas metodas, vizualizacijos galimybių šiandienai galima rasti pačių įvairiausių.

Pačia bendriausia prasme vizualizaciją galima skirstyti į meninę vizualizaciją (nuotraukos, paveikslai, įvairiausios reklamos formos ir t.t.) ir informacijos perteikimui skirtą vizualizaciją. Jau

pakankamai seniai vizualizuojamus objektus stengiamasi perteikti kuo patraukliau, todėl informacijos perteikimui skirtos vizualizacijos būna ne tik informatyvios, bet ir patrauklios savo išvaizda, ypač tuomet kai vizualizacija kuriama verslo procesams plėsti, pritraukti naujiems užsakovams ir pan. Todėl kalbant apie informacijos perteikimui skirtas vizualizacijas meninės jų pusės taip pat negalima nuvertinti.

Kadangi šis darbas yra skirtas informacijos perteikimo vizualizacijų sistemų analizei, toliau kalbant apie vizualizaciją, ji bus suprantama kaip informacijos perteikimo vizualizacija, jei nepaminėta kitaip.

Taigi, kokios gali būti informacijos perteikimo vizualizacijos? Vienareikšmiškai suklasifikuoti vizualizacijas gana sunku, nes daugelyje skirtingų šaltinių jos skirstomos šiek tiek skirtingai, tačiau vyrauja bendra tendencija [1] [2] [3] [4], kuria pasinaudodamas šiame darbe vizualizacijos sistemas suskirsčiau į

- Informacijos vizualizacija (*angl.information vizualisation*),
- Žinių vizualizacija (*angl. knowlenge vizualisation*),
- Mokslinė vizualizacija (*angl. scientific vizualisation*).

Kaip minėjau, galima klasifikuoti ir kitaip, tačiau šios vizualizacijos rūšys yra geriausiai išskiriamos todėl jas ir apibūdinsiu. Kuo šios vizualizacijos rūšys skiriasi?

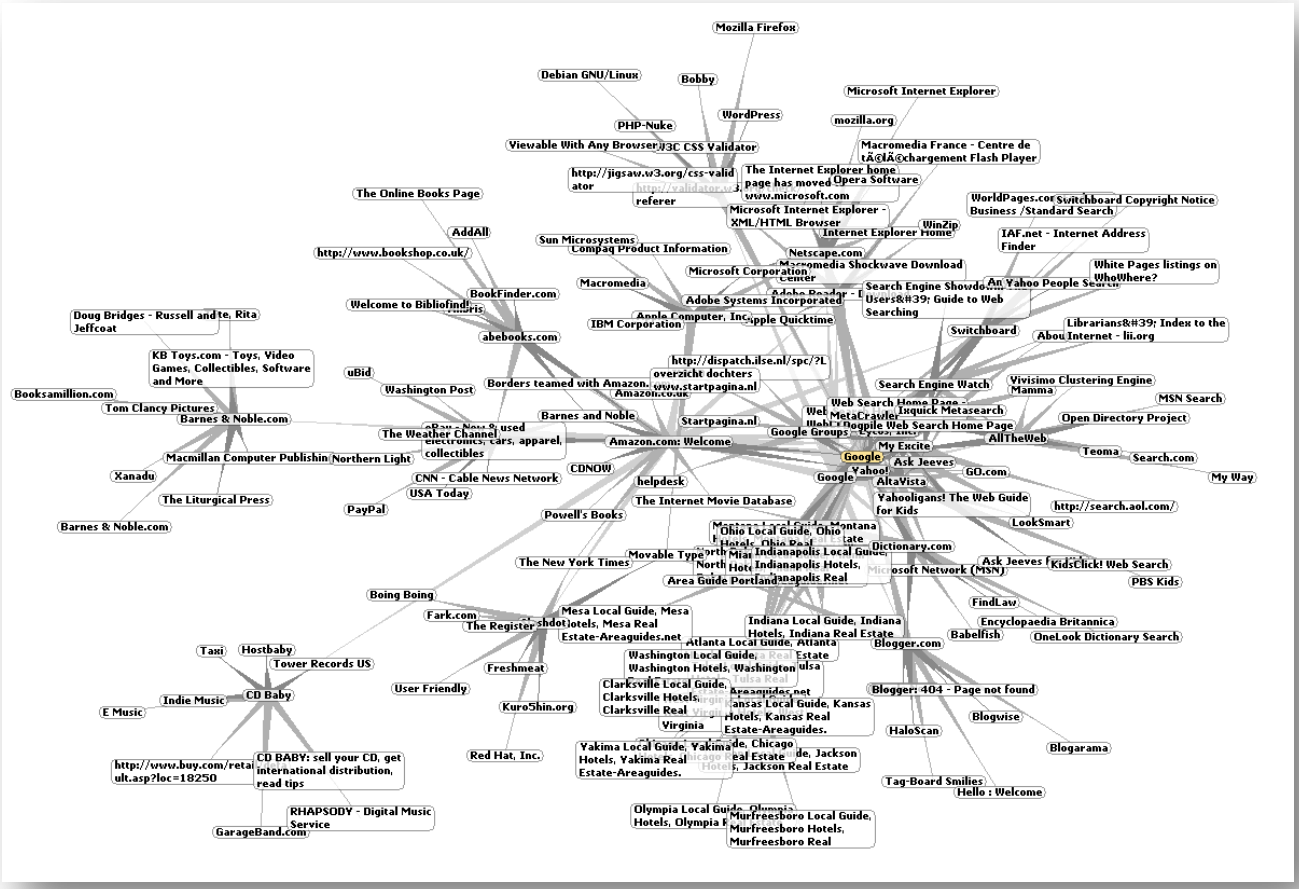
Informacijos vizualizacija [1] yra skirta konkrečios informacijos perteikimui vizualiai: failų struktūros atvaizdavimas, žemėlapiai ir pan. – tai yra didelės apimties „nenumeruojama“ informacija. Pav. 1 pateiktas informacijos vizualizacijos pavyzdys – pasaulinio žiniatinklio struktūra supanti paieškos sistemos „Google“ šerdį. Terminas informacijos vizualizacija „galėtų būti naudojamas nusakant visus darbus duomenų vizualizacijos, informacijos grafikos, žinių vizualizacijos, mokslinės vizualizacijos ir vizualaus dizaino srityse“[1].

Žinių vizualizacija [3] [4] yra skirta žinių, išreikštų vizualine reprezentacija, perdavimui tarp mažiausiai dviejų žmonių (arba tarp proceso ir žmogaus/žmonių grupės). Tikslas – pagerinti žinių perdavimą panaudojant ir kompiuterinį, ir ne kompiuterinį vizualizacijos kūrimo metodus drauge. Tokių vizualizacijų pavyzdžiais yra:

- Eskizai
- Diagramos
- Paveikslai
- Įvairūs objektai

[1] The term Information visualization could be taken to subsume all developments in data visualization, information graphics, knowledge visualization, scientific visualization and visual design.

- Interaktyvios vizualizacijos
- Vaizduotė
- Pasakojimai



Pav. 1 Pasaulinio žiniatinklio struktūra supanti „Google" paieškos sistemą [17]

Mokslinė vizualizacija [2] yra tarpdisciplininė mokslo šaka, kurios pagrindinis uždavinys yra trijų matmenų reiškiniai – tokie kaip architektūrinės, meterologinės, medicininės, biologinės sistemos.[2] Mokslinė vizualizacija koncentruojasi į kompiuterinės grafikos panaudojimą kuriant paveikslus (vizualizacijas) tam, kad būtų galima lengviau suprasti/pateikti didelės apimties, sudėtingus mokslinių tyrimų ar skaičiavimų rezultatus.

Taigi, iš apibrėžimų seka, kad visos vizualizacijos rūšys vienaip ar kitaip tarpusavyje yra glaudžiai persipynusios ir negali būti vienareikšmiškai išskiriamos. Šiame darbe bus nagrinėjama tik

[2] Scientific visualization (also spelled scientific visualisation) is an interdisciplinary branch of science, primarily concerned with the visualization of three dimensional phenomena, such as architectural, meteorological, medical, biological systems.

interaktyvioji vizualizacija – vienas iš būdų perteikti žinias, plačiąja prasme – informaciją interaktyviai.

Interaktyvioji vizualizacija

Interaktyviosios vizualizacijos yra kompiuterinės vizualizacijos, kurios leidžia naudotojui prieiti, kontroliuoti, manipuliuoti ar kombinuoti įvairaus tipo informacija. Tokių vizualizacijų pavydžiais yra

- **Pramonės procesų vizualizacija** – tam yra sukurta nemažai programinės įrangos paketų ar net programų šeimų, leidžiančių standartizuotais metodais kurti vizualizacijos projektus, surinkti ir apdoroti duomenis.
- **Medicinos realaus laiko tyrimų** (pvz. ultragarso) – tam yra kuriama specializuota programinė yranga konkrečiam procesui vizualizuoti. Šioje srityje praktiškai nesutinkama jokių standartinių paketų – visa programinė įranga yra specializuota.
- **„Protingo namo" vizualizacijos ir valdymo** – šiuo metu ypač suintensyvėjusi vizualizacijos sistemų kūrimo PĮ gamyba.
- **Kitos sritys** – realiai bet kuri vizualizuojama sritis, kuri pateikia vartotojui duomenis apie sistemą/procesą ir leidžia jais manipuliuoti.

Interaktyvios vizualizacijos skirtingai nuo statinės informacijos leidžia daug geriau pritraukti naudotojo dėmesį (kas ypač svarbu vizualizuojant sudėtingus ir kritines pasekmes klaidos atveju nešančius procesus).

Interaktyvios vizualizacijos sistemos ir jas sudarantys elementai

Vizualizacijos sistema – tai specializuota (vienetinė) programinė įranga arba specializuotų programų paketo pagalba sukurta vizualizacija, skirta konkretiems duomenims ar procesams vizualizuoti. Kitaip tariant, vizualizacijos sistema – tai programinis produktas skirtas konkrečiam tikslui atvaizduoti, sukurtas arba standartinėmis priemonėmis, arba specializuoto programinės įrangos, skirtos vizualizacijos sistemoms kurti, pagalba. Tokios specializuotos programinės įrangos pavyzdžiais yra šie programinės įrangos paketai:

- VijeoCITECT
- Monitor PRO
- Wonderware InTouch
- ir kiti

Kiekviena vizualizacijos sistema susideda iš elementų. Tam, kad geriau suprasti elemento sąvoką, palyginsiu ją su įprastos programinės įrangos elementais.

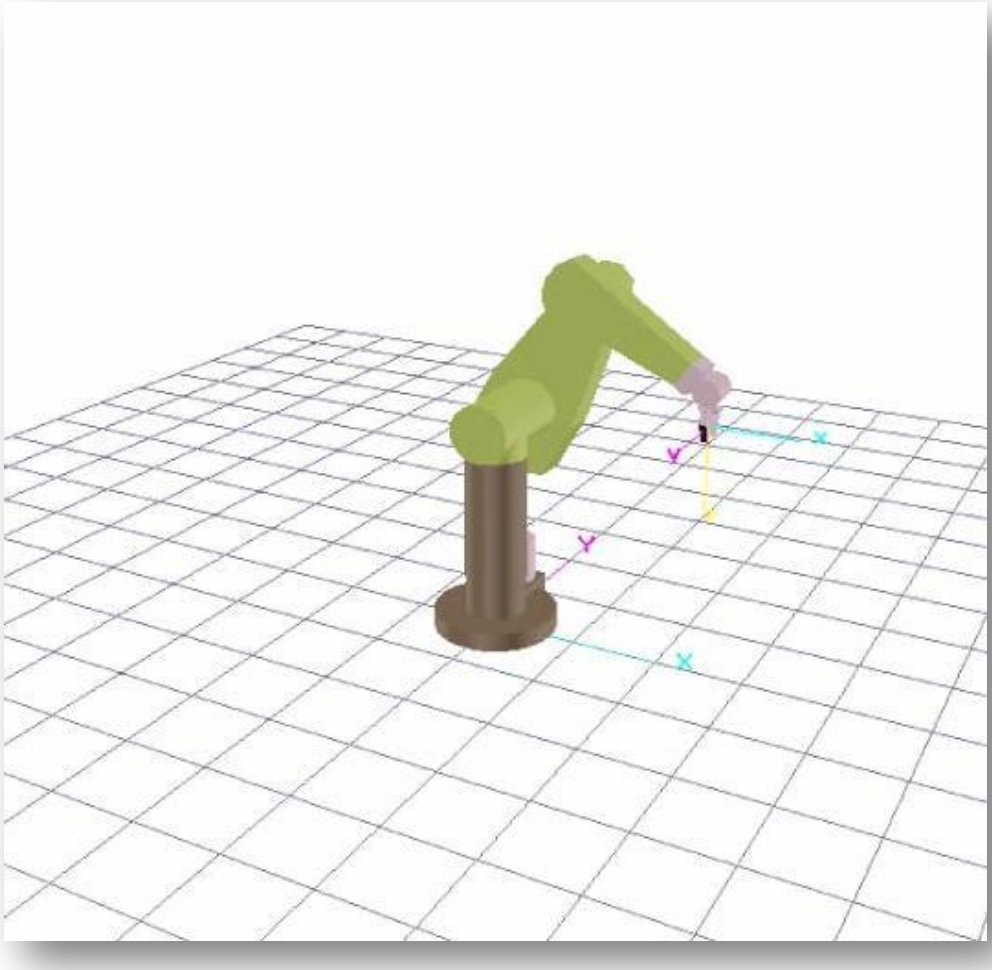
Įprastoje programinėje įrangoje naudojami naudotojo sąsajos komponentai (*angl. user control*). Tai yra mygtukai, meniu punktai, teksto įvedimo laukeliai, iškrentantys sąrašai ir t.t. Vizualizacijoje šie elementai taip pat gali būti naudojami, tačiau vizualizacijos sistemose atsiranda ir šiek tiek kitokios paskirties elementų. Tarkime, vizualizacija yra skirta „protingo namo“ valdymui. Tokioje vizualizacijoje dažniausiai valdomi dalykai yra

- Apšvietimas
- Temperatūra
- Žaliuzės/garažo vartai
- Vėdinimas

Jeigu mes norime pateikti naudotojui informaciją apie temperatūrą patalpoje, ji paprastai pateikiama elemente, skirtame temperatūros stebėjimui, nustatymui ir šildymo režimo pakeitimui. Tai reiškia, jog šis elementas yra sudėtinis, kurį sudaro standartinės programinės įrangos elementai (mygtukai, slankjuostės, paveikslėliai ir t.t.). Taigi, vizualizacijos elementą galima apibūdinti kaip komponentų rinkinį (atskirais atvejais ir vieną komponentą), skirtą konkrečiam procesui/duomenims atvaizduoti.

Trimatis vizualizacijos elementas – tai toks elementas, kuris vizualizacijos sistemoje priklausomai nuo naudotojo poreikių gali būti apžiūrimas iš bet kurio erdvės taško aplink jį – t.y. naudotojas pats gali pasirinkti, kaip elementas atrodo ekrane. Pavyzdžiui tarkime, jog turime vizualizacijos sistemą, skirtą atvaizduoti mechanizuotos rankos darbui (Pav. 2). Dvimatėje erdvėje tokios rankos darbą atvaizduoti praktiškai neįmanoma. O panaudojus trimatį elementą jis tampa ne tik įmanomas, bet ir labai vizualus bei paprastas. Pav. 2 pateiktas trimačio elemento pavyzdys.

Vienas iš šio darbo tikslų yra išsiaiškinti, ar trimačių elementų panaudojimas vizualizacijos sistemose atneša papildomos informacijos/paprasumo naudoti naudotojui, ar sukelia papildomų problemų. Tam buvo sukurta vizualizacijos sistema, kuri yra realiai sukurtos, įdiegtos ir eksploatuojamos vizualizacijos dalis, naudojanti trimačius elementus vizualizuojamam objektui ir duomenims perteikti.



Pav. 2 Mechanizuotos rankos vizualizacija – trimatis vizualizacijos elementas [16]

Keliama problema ir projekto tikslas

Dabartinių plačiai rinkoje naudojamų vizualizacijos sistemų funkcionalumas ir teikiamos galimybės daugiau ar mažiau yra panašios, vyrauja ta pati koncepcija tiek duomenų surinkimo, tiek pateikimo atžvilgiu. Sukurtos sistemos skirtumas nuo šiuo metu rinką užvaldžiusių sistemų yra trimačių elementų naudojimas. Tokios galimybės iškėlimas kaip realiai pritaikomos praktikoje ir yra pagrindinis idėjos variklis. Pasinaudojus trimačiais elementais taip pat norima padėti spręsti vieną iš besikartojančių praktikoje pasitaikančią problemą – operatyvų supratimą apie avarinę situaciją sukėlusią priežastį ar objektą.

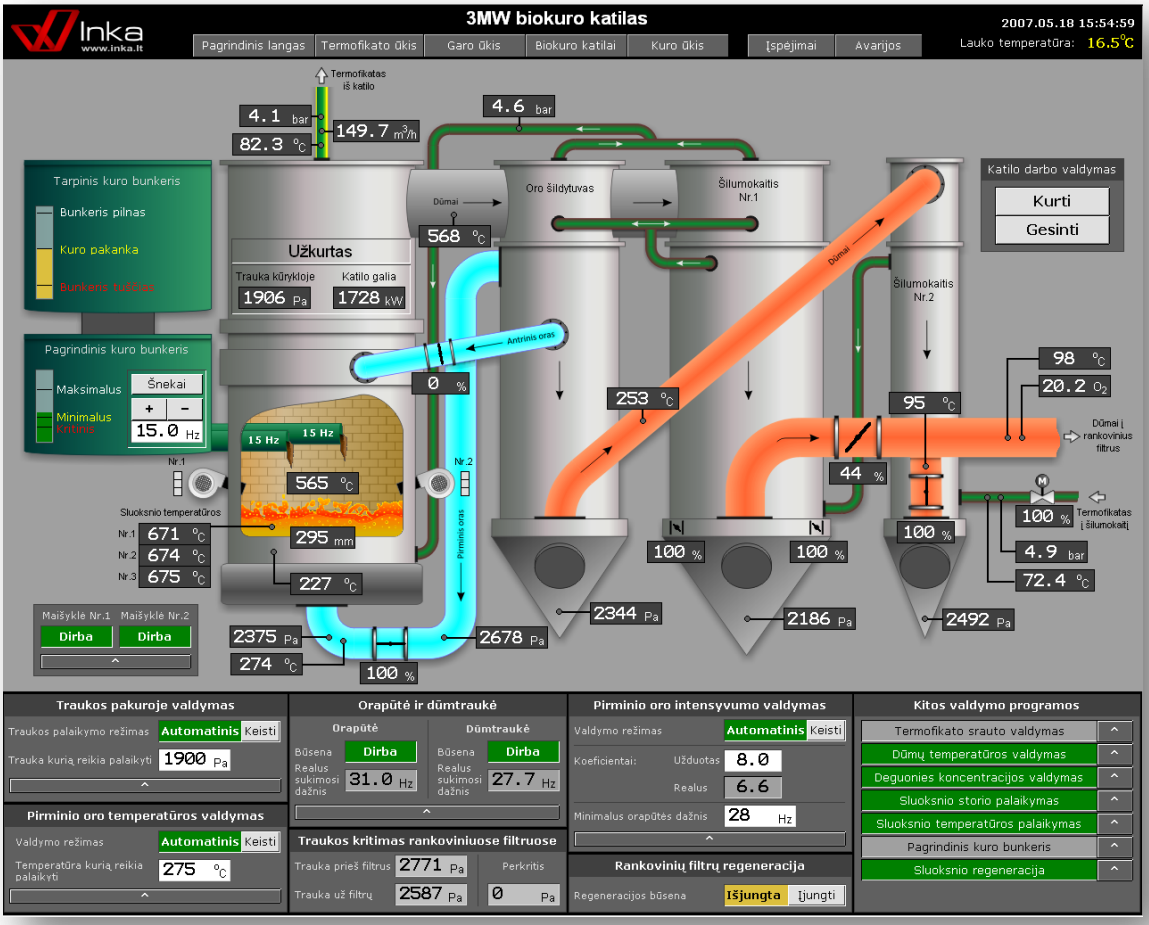
Negalima pamiršti, jog vienas lemiamų veiksnių pristatant produktą vartotojams yra estetinis vaizdas. Pastaruoju metu programiniai produktai, skirti vizualizavimui ar plačiam vartojimui yra kuriami taip, kad pirmiausiai trauktų akį, ne tik tenkintų tikruosius sistemai keliamus reikalavimus.

Norint kurti konkurencingą produktą, gyvybiškai būtina tai įvertinti bei imtis atitinkamų priemonių norimam rezultatui pasiekti.

Projekto tikslas yra sukurti demonstracinę interaktyvios vizualizacijos sistemą, kuri leistų išskirti trimačių elementų naudojimo vizualizacijos sistemos teigiamas ir neigiamas savybes. Taip pat maksimaliai pritaikyti kuriamą sistemą prie tokiai vizualizacijos sistemai keliamų reikalavimų: maksimaliai supaprastinti trimačių objektų panaudojamumą bei ištirti, ar sistema, naudojanti trimačius komponentus yra pakankamai stabili ir saugi juos naudoti. Ši sistema turėtų būti kiek galima paprastesnė, be to, glaudžiai tarpusavyje susieti trimatės ir dvimatės grafikos elementus tam, kad būtų išvengta bereikalingų pastangų kuriant trimačius elementus kur pilnai užtektų jų dvimačių analogų.

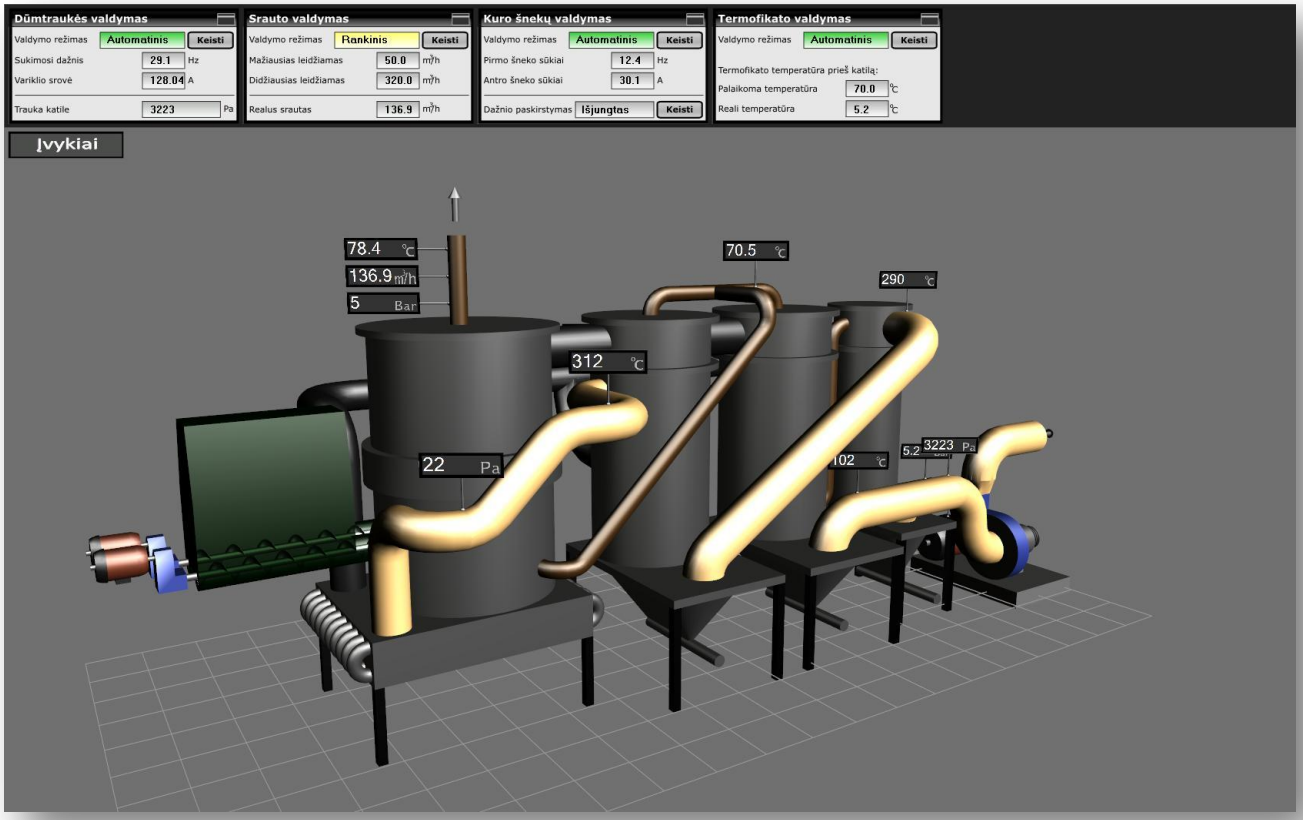
Pagrindinis kuriamos sistemos tikslas – išsiaiškinti, ar tokio pobūdžio kuriama sistema yra efektyvi operatoriaus darbo palengvinimui. Taip pat suteikti galimybę kurti gražesnes ir patrauklesnes vizualizacines sistemas, kas, kaip jau buvo minėta, kuo toliau tuo labiau tampa prioritetiniu dalyku.

Kam reikalingi trimačiai elementai? Paprastai, daugumoje vizualizacijos sistemų, kad ir kokios jos būtų sudėtingos, vizualizuojamiems procesams perteikti paprastai užtenka vien dvimačių elementų, išskyrus tam tikrus specifinius atvejus (pvz. robotizuotos rankos automobilių surinkimo linijose ir pan.). Tačiau net ir sukūrus dvimatę vizualizacijos sistemą kartais tampa labai sunku vienareikšmiškai ir suprantamai perteikti informaciją apie procesą. Problema atsiranda tuomet, kai informacijos yra pakankamai daug, o atvaizduojamas objektas (šildymo katilas, konvejeris ir pan.) yra santykinai mažas palyginus su atiduodamos informacijos kiekiu. Kaip pavyzdį pateiksiu vandens šildymo katilą (Pav. 3). Kaip matyti iš pateikto paveikslėlio, informacijos įrenginys atiduoda tikrai labai daug: įvairiausių katile esančio šildomo vandens zonų temperatūros, slėgiai, kuro padavimo sistemos parametai, degimo proceso temperatūros, oro srautų parametrai ir t.t.



Pav. 3 Realiai veikianti dvimatė vandens šildymo katilo vizualizacija

Natūralu, kad visi davikliai bei išvedžiojami vamzdynai nėra vienoje katilo pusėje, o proporcingai išdėlioti visur aplink įrenginį. Problema atsiranda tuomet, kai didelį kiekį informacijos nešėjų (daviklių), išdėliotų trimatėje erdvėje reikia transformuoti į dvimatį analogą kompiuterio ekrane. Tuomet tenka iškreipti realų įrenginio vaizdą, daviklius sudėlioti ne ten kur jie realiai yra, bet ten kur jie telpa ekrane. Čia ir iškyla viena pagrindinių vizualizacijų, naudojančių dvimačius elementus, problema: informacija pateikiama neadekvati realybei, vienu metu matomos informacijos kiekis tampa labai didelis ir to pasekoje naudotojui tampa sunku greitai susigaudyti vykstančiame procese. Ši problema ypač paaštrėja iškilus ekstremaliai situacijai (sugedus sistemą aptarnaujantiems būtiniems įrenginiams, ko pasekoje iškyla perkaitimų, sprogimų, įrangos sugadinimo ir t.t. pavojus) kai reikia greitai ir vienareikšmiškai nustatyti sutrikimo vietą, kad būtų galima kuo greičiau pašalinti sutrikimo priežastis. Pav. 4 pateiktas šio darbo metu sukurtas trimatis vandens šildymo katilo vizualizacijos analogas.



Pav. 4 Realiai veikiančios vizualizacijos trimatis analogas

Kaip trimačiai elementai gali padėti spręsti išvardintas problemas? Naudojant trimatį objektą, nebelieka būtinybės vizualizuojamą objektą transformuoti į dvimatį – jį galima atvaizduoti tiksliai tokį, koks jis realiai yra be jokių iškraipymų. Daviklius atvaizduoti tokiame modelyje taip pat galima tiklsliai, nebijant perkrauti naudotoją informacija tuo pačiu metu pateikiant ją visą. Vienu metu ekrane matoma tik ta informacijos dalis, kuri yra šiuo metu atsukta į naudotoją. Visa kita informacija lieka „paslėpta“ už modelio ir tampa matoma modelį apsukus.

Egzistuojantys sprendimai

Plačiajai rinkai siūlomų vizualizacijos paketų, kurie palaikytų trimačių objektų naudojimą, yra labai nedaug. Daug projektų, kuriuose yra panaudoti trimačiai objektai, arba net visas projektas realizuotas trimatėje erdvėje, yra sukurti specialiai konkrečiam objektui vizualizuoti, t.y. jie yra vienetiniai. O SCADA sistemų, kurios turėtų savyje integruotą trimačių objektų biblioteką, kuria būtų galima laisvai manipuliuoti, yra vos keletas. Kaip minėta, taip yra dėl to, kad yra pakankamai sunku realizuoti IDE (Integrated Development Environment) aplinką, kuri leistų laisvai manipuliuoti

trimačiais objektais. Tarp populiariausių sprendimų galima būtų paminėti dvi SCADA sistemas, palaikančias trimačius objektus: *AdAstrA Research Group, Ltd* projektą *TRACE MODE* ir kompanijos *VRcontext* sukurtą SCADA sistemą pavadinimu *Walkinside*.

„Walkinside“ sprendimas

Kompanijos VRcontext vizualizacijos sistema *Walkinside* daugiau skirta sudėtingų pramoninių procesų simuliacijai, turint galimybę stebėti procesą realiuoju laiku. Tai sudėtingas vizualizacijos paketas, leidžiantis apdoroti modelį, kurį sudaro iki 10^9 poligonų. Jame panaudotas laisvo kodo grafinis variklis OpenGL [5], kuris gali būti naudojamas ne tik Microsoft Windows bet taip pat ir kitose opeacinėse sistemose, tokiose kaip Linux, MacOS ar Solaris.

Sistema neskirta modelių kūrimui ar generavimui – ji tik gali paversti CAD sistemų sukurtus modelius trimačiais objektais. Sistema naudoja dvi technologijas, leidžiančias trimatį modelį paverti tikroviškesnį ir natūraliai suprantamu:

- *Real-Time Collision Detection* technologija, ir
- *Gravity simulation* technologija

Real-Time Collision Detection skirta visų modelyje egzistuojančių judančių objektų tarpusavio susidūrimų (kolizijų) radimui, kas leidžia tikroviškai matyti bei vertinti aplinką, kurioje veikia judantis objektas (žmogus, procesą aptarnaujantys įrenginiai ir pan.)

Gravity simulation technologija taip pat skirta tam, kad modelis atrodytų kuo realistiškiau. Jos pagalba imituojama natūrali gravitacija, kurios pagalba modelyje judantys objektai linkę „kristi ant grindų“.

„Walkinside“ privalumai ir trkumai

Vienas iš jau minėtų *Walkinside* privalumų yra tas, kad ši vizualizacijos sistema leidžia naudoti modelius, kurie yra gana dideli – sudaryti iš iki milijardo poligonų. Ši sistema palaiko automatinį modelių įkėlimą iš CAD, lazerinio skenavimo ir kitų šaltinių, kas labai paspartina ir supaprastina darbą. Intuityvi ir ganėtinai paprasta vartotojo sąsaja. Nedideli reikalavimai techninei įrangai leidžia sistemai veikti greitai ir efektyviai.

Pagrindinis šios sistemos trūkumas (kaip ir daugumos trimačiais objektais manipuliuojančių sistemų) yra tas, kad iš vartotojo yra reikalaujama specifinių žinių norint efektyviai išnaudoti sistemos teikiamas galimybes. Intuicijos, susipažinimo su dokumentacija ir nedidelės patirties neužtenka norint realizuoti projektą.

„TRACE MODE“ sprendimas

TRACE MODE 6.0 galima būtų apibūdinti kaip klasikinę vizualizacijos sistemą su gal plačiai išplėstomis galimybėmis. Ji ne tik turi visas standartinės SCADA sistemos funkcijas, bet kartu palaiko trimačius objektus. Naudojamas grafinis variklis – taip pat OpenGL. Palaiko „daugiasluoksnes“ scenas, kas leidžia tame pačiame ekrane pateikti kelias skirtingas technologines posistemes. Unikali *3D Fast+* technologija leidžia efektyviai pateikti duomenis vartotojui.

„TRACE MODE“ privalumai ir trūkumai

Jeigu vizualizacijos sistemoje naudojamas pertekliškumas (arba dubliavimas saugumo dėlei), tai paprastai toks pertekliškumas būna dviejų lygmenų. *TRACE MODE* 6.0 vizualizacijos paketas taip pat palaiko pertekliškumą, tačiau jis gali būti ne tik dviejų, bet ir trijų lygmenų.

TRACE MODE, taip pat kaip ir *Walkinside* nesprendžia problemos, kaip trimatę grafiką glaudžiai komponuoti kartu su paprasta dvimate grafika nenaudojant trimatės grafikos variklio. Dvimatės grafikos technologijos, tokios kaip Microsoft GDI ar GDI+ yra žymiai stabilesnės, daug mažiau reiklios kompiuteriniams resursams bei paprasčiau taikomos ir suprantamos. Todėl šių technologijų panaudojimas sumažintų viso produkto svorį [6]. Paprastesnės vizualizacijos sistemos, kurios nėra perkrautos įvairių technologijų mišiniu, turi nemažai privalumų. Kuo daugiau funkcionalumo, tuo sistema „sunkesnė“, todėl projektuojant vizualizacijos sistemą reikia ne maksimaliai plėsti jos funkcionalumą, bet rasti kokybišką santykį tarp funkcionalumo ir svorio.

Įgyvendinimo problemos

Kuriant vizualizacijos sistemą neišvengiamai kyla būtinybė pasirinkti vizualizacijos tipą ir jos architektūrą. Vizualizacijos sistemos paprastai būna šių tipų:

- *Standalone* – tai toks vizualizacijos variantas, kai ir duomenų surinkimo serveris, ir jų apdorojimas bei pateikimas vizualizacinei daliai bei pati vizualizacinė dalis yra viename kompiuteryje.
- *Client-server* arba *distributed* – kada duomenys yra tinkle nutolusiam serveryje, o vizualizacija atliekama kitame kompiuteryje, kur vizualizacija gali būti tiek paprasčiausia aplikacija, tiek per WEB sąsają organizuota aplikacija.

Pasirinkus *standalone* architektūrą, SCADA sistemos kūrimas, kuri palaikytų trimačius objektus, nėra labai sudėtingas uždavinys. Nekeliami dideli reikalavimai vizualizacijos aplinkai perduodamam duomenų kiekiui, vizualizacija gali vykti raliuoju laiku su gana nedideliais laiko

vėlinimais. Pačių trimačių elementų kurimui yra pakankamai paprastų ir gerai išdirbų priemonių, tokių kaip OpenGL ar MS DirectX. Naudojant MS Windows Vista operacinę sistemą, trimačių objektų kūrimui galima panaudoti naują Microsoft technologiją WPF (Windows Presentation Foundation), kuri turi plačias galimybes kurti trimačius objektus [7], be to yra pilnai integruota į operacinę sistemą, todėl labai gerai su ja suderinama. Tačiau naudojant šį principą atsiranda būtinybė visą sistemą „nešiotis“ ten, kur yra poreikis turėti vizualizaciją.

Client-server architektūra yra geresnė tuo, kad yra tik vienas (sąlyginai – gali būti ir daugiau) duomenų šaltinis, prie kurio gali jungtis daugiau nei vienas klientas. Jei vizualizacijos aplikacija yra organizuota per WEB sąsają, yra keletas būdų ją išnaudoti trimačiams objektams pateikti [8]. WEB sąsaja gera tuo, kad vizualizacijos sistemos klientas tampa gana kompaktiškas ir nesudėtingas dėl tam tikrų WEB sąsajos apribojimų.

Pasirinkus architektūrą ir projektuojant sistemos karkasą, patartina atsakyti į šiuos klausimus: kas turėtų būti padaryta su duomenimis iki kol jie bus pateikti vizualizacijos sistemai ir kas gali būti padaryta po duomenų pateikimo vartotojui. Duomenų apdorojimo (spaudimo, atmetimo dėl kokybės praradimo ir pan.) ir pateikimo vartotojui algoritmai yra kertinis akmuo kuriant vizualizacijos sistemą, todėl ir yra taip svarbu juos projektuoti taip, kad būtų užtikrinta galimybė duomenis apdoroti prieš juos pateikiant vartotojui ir veiksmai su duomenimis po jų pateikimo [9]. Kartais duomenų, pateikiamų sistemai yra tiek daug, kad dauguma iš jų neša nebe papildomą, o perteklinę informaciją. Tokiu atveju būtini efektyvūs algoritmai jų spaudimui, diskretizavimui ar filtravimui. Tai būtina daryti dėl to, kad pertekliniais duomenimis apkraunant sistemą, prarandama greitaveika, padidėja stabilumo sutrikimo tikimybė ir išauga kompiuterinių resursų naudojimas – o tai nėra teigiami dalykai kokybiškai vizualizacijos sistemai.

Priklausomai nuo vizualizuojamų duomenų apimties, gali tekti rinktis tokią sistemos architektūrą, kuri leistų pakankamai greitai duomenis apdoroti ir pateikti vartotojui. Ne visos architektūros yra vienodai pajėgios susitvarkyti su dideliais duomenų kiekiais arba per greitais jų atnaujinimais, todėl būtina iš anksto numatyti, kokiose ribose vizualizacija veiks [10].

Jei reikalingų duomenų kiekis yra pakankamai didelis, arba yra daug vienu metu vizualizuojamų objektų, atsiranda kita problema – visų reikalingų duomenų pateikimas iš karto. Tam vieno monitoriaus skiriamosios gebos gali neužtekti, todėl yra kuriami sprendimai, kaip prijungti kelis monitorius prie vienos veikiančios sistemos [11]. Paprastai tokiais atvejais kelių monitorių sistema veikia kaip vienas monitorius su didele skiriamąja geba. Tačiau pagrindinė tokios sistemos problema yra ta, kad grafiniai procesoriai, kurie palaiko daugiau nei du monitorius vienu

metu paprastai neturi aparatiško trimatės grafikos spartinimo, o tai reiškia, kad vizualizacijai, kuri palaiko trimačius objektus, šis problemos sprendimo būdas netinka.

Trimačių objektų kūrimas

Trimačių objektų kūrimas nėra paprastas dalykas – jis reikalauja tiek specifinių vartotojo įgūdžių, tiek ir specialių įrankių, kurie paprastai yra labai brangūs. Be to, CAD sistemoje (pvz., AutoCAD) sukurtų objektų formatas paprastai turi būti pakeistas į tokį, kurį palaiko vizualizacijos sistemoje naudojamas trimatės grafikos variklis. Tam taip pat reikia ir papildomų vartotojo žinių bei įgūdžių ir specialių įrankių. Papildomų sunkumų kelia ir tai, jog trimačiams objektams kurti paprastai naudojama ne viena, bet kelios CAD sistemos dėl to, kad kažkuri konkreti sistema neturi reikiamo funkcionalumo ir panašiai.

Korekcijos problemos

Kaip minėta, tam, kad vizualizacijos variklis atpažintų objekto formatą, būtina tą formatą pritaikyti. To pasekoje gali tapti nebeįmanoma objekto redaguoti, nes CAD sistema gali nebeatpžinti naujojo formato. Todėl būtina ne tik saugoti bibliotekoje talpinamą objekto variantą, bet ir turėti jo originalą. Kita problema kyla tuomet, kai objektas yra ne sukurtas, bet gautas iš kitų šaltinių (interneto, pirktas ir pan.). Tuomet jo pakoreguoti tampa praktiškai neįmanoma, arba reikia vizualizacijos sistemoje kurti sudėtingą posistemę, leidžiančią koreguoti bibliotekoje turimus trimačius objektus. Tai būtų labai neefektyvu, nes tokios posistemės sukūrimas labai brangus bei sudėtingas, be to egzistuoja gana nemažai sukurtų įrankių, tarp jų ir CAD sistemos, turinčių šį funkcionalumą. Vienas iš šių problemų sprendimo variantų būtų naudoti standartinį trimačių objektų formatą, kurį palaiko dauguma kitų sistemų.

Stabilumo problemos

Kadangi trimatė grafika ypatingai reikti kompiuteriniams resursams, atsiranda sistemos stabilumo ir veikimo patikimumo problemos. Sistemos stabilumui užtikrinti tampa būtina tam tikrais laiko tarpais perkrovinėti sistemą tam, kad neužterštų atminties iki tokio lygmens, kuris sukelia duomenų praradimą ir sistemos lūžį. Taip atsitinka todėl, kad trimatė grafika naudoja didžiulius kiekius grafinės ir operatyvinės atminties sparčiam savo darbui užtikrinti. Tai ir priveda prie daug spartesnio atminties užteršimo.

Skirtingų operacinių sistemų palaikymas

Kuriant multiplatforminę vizualizacijos sistemą, atsiranda labai dideli apribojimai naudojamiems įrankiams. Didžiausią keblumą kelia trimatės grafikos variklio kūrimas, nes migruojant iš vienos operacinės sistemos į kitą atsiranda visa eilė nesuderinamumo problemų, tokių kaip atskirų įrenginių tvarkyklės (ypač grafinių kortų) ir panašiai. Be to, kuriant grafinį variklį, jis optimizuojamas tiek operacinės sistemos atžvilgiu, tiek ir naudojamos aparatūros atžvilgiu. Todėl kyla problemų ne tik migruojant tarp operacinių sistemų, bet ir tarp skirtingų kompiuterių. Yra tam tikri metodai, kurie leidžia užtikrinti didesnį suderinamumą, bet tai yra daroma papildomai išnaudojant kompiuterinius resursus, kas automatiškai atsiliepia visos sistemos stabilumui ir veikimo greitaveikai.

Kitas aspektas, į kurį reikia atsižvelgti kuriant multiplatforminę vizualizacijos sistemą (ir benrdai visą programinę įrangą) yra tas, jog gali būti naudojama toks SDK paketas, kuris yra multiplatforminis (java, flash ir pan.) Kaip žinoma, technologijos, kurios yra pritaikytos visoms ar bent jau daugumai operacinių sistemų, vistiek yra orientuojamos daugiau į kažkurią vieną, kurioje yra ir stabiliausios, ir greičiausiai dirba. Vizualizacijos sistemos (ypač realaus laiko procesų) yra ypač reiklios greitaveikai, todėl patikimiausiai ir greičiausiai veikia naudojant konkrečią operacinę sistemą. Jeigu įprastai programinei įrangai kurti yra bent keletas multiplatforminių technologijų, tai trimatės grafikos kūrimui yra vienintelis OpenGL paketas. Todėl reikia įvertitnti, ar tikrai verta kurti multiplatforminę vizualizacijos sistemą.

Šis projektas sukurtas vienai konkrečiai operacinei sistemai Microsoft Windows XP. Kadangi projekto tikslas yra trimačiai elementai bei jų panaudojimas vizualizacijos sistemose, todėl pasirinkau operacinę sistemą, kurioje šį tikslą pasiekti paprasčiausia, OS yra paplitusi labiausiai ir yra daug tokios vizualizacijos sistemos kūrimą palengvinančių įrankių.

Trimatės grafikos kūrimo priemonės

Kaip minėta, yra du pagrindiniai API, skirti trimatės grafikos programavimui: OpenGL ir Microsoft DirectX. Rinkoje yra labiau paplitusi Microsoft technologija DirectX, ypač žaidimų kūrime ir tose sistemose, kuriose reikalaujamas rimtas aparatūrinis spartinimas. OpenGL sistema plačiau naudojama CAD sistemose, mokslinei vizualizacijai, skrydžių simuliacijai. Renkantis priemonę, su kuria bus kuriama vizualizacijos sistema reikia išskirti ir gerai išanalizuoti pagrindinius šių API privalumus ir trūkumus.

OpenGL

Teigiamos savybės:

- Galima naudoti įvairiose operacinėse sistemose
- Gerai išnaudoja pritaikytos aparatūros spartinančiąsias galimybes
- Turtinga funkcijų biblioteka
- Palaiko praplečiamumą (*extentions*)
- Nemokamas
- Labai stabilus

Neigiamos savybės:

- Sudėtingas funkcijų panaudojimas
- Sunkokai randami sudėtingesnės užduoties pavyzdžiai
- Ne visos grafinės kortos palaiko aparatinį OpenGL spartinimą
- Paprastesniam panaudojimui būtinos papildomos bibliotekos
- Sudėtingesnis įvesties įrenginių naudojimas

DirectX

Teigiamos savybės:

- Paprastas naudojimas
- Daug funkcijų, skirtų darbo supaprastinimui
- Labai gerai išnaudoja pritaikytos aparatūros spartinančiąsias galimybes
- Didžioji dauguma grafinių kortų palaiko aparatūrinį DirectX spartinimą
- Integruotas į MS operacines sistemas
- Yra daug patogių programavimo įrankių, nesudėtingai palaikančių DirectX bibliotekas
- Labai daug dokumentacijos ir pavyzdžių
- Platus naudojamumas, kas verčia nuolatos tobulinti DirectX
- Palaiko daugiau nei vieną grafikos interfeisą (multiple graphics adapters)
- Yra CAD sistemų, kurios turi failų formatą, tiesiogiai pritaikytą DirectX aplinkai
- Patogiai palaiko daug skirtingų įvesties įrenginių
- Nemokamas

Neigiamos savybės:

- Palaiko tik vieną operacinę sistemą (MS Windows)
- Daug skirtingų interfeisų

Šiuo metu didžioji dauguma vizualizacijos sistemų yra pritaikytos dirbti MS Windows operacinėje sistemoje todėl, kad ši OS yra mokama ir dėl to pilnai palaikoma bei pastoviai atnaujinama. Be to, MS Windows operacinei sistemai yra pritaikyta labai daug įrankių bei programų, papildomai reikalingų vizualizacijos darbui užtikrinti.

Vizualizuojami procesai

Šios sistemos pagrindinis tikslas yra ne paversti vizualizacijos sistemą patrauklesnę ar gražesnę, bet sudėtingų objektų, kurie keičia padėtį erdvėje daugiau nei viena kryptimi arba pateikia labai daug informacijos, vizualizacijos galimybėms išplėsti. Yra daug skirtingų procesų, kuriuose dalyvauja mašinos-robotai, ir jie gali judėti erdvėje laisvai bei įvairiomis kryptimis. Tokiu atveju standartinių vizualizacijos sistemų galimybių paprastai arba neužtenka, arba būna pakankamai sunku ir neefektyvu kokybiškai padaryti proceso vizualizaciją. Keletas procesų pavyzdžių, kur ši vizualizacijos sistema būtų taikoma:

- įvairios gamybos linijos
- nestandartiniai transporteriai, (rotaciniai kuro pergabenimo, ...)
- pozicionavimas pastatuose (signalizacijos sistemos, aktyvių patalpų stebėjimas, ...)
- ir kiti

Kadangi vizualizacijos sistemos dažniausia kuriamos užsakovo verslo poreikiams tenkinti, trimačių objektų funkcionalumas privalo maksimaliai tenkinti užsakovo poreikius [12], kiek įmanoma adaptuojant komponentus taikomai sričiai. Todėl kyla poreikis analizuoti tam tikras verslo kryptis (gamyba, medicina, vandenvala, mokslas, ...) ir atitinkamai adaptuoti komponentus pasirinktai sričiai. Kadangi kiekviena iš paminėtų sričių yra labai specifinė su savitais reikalavimais greitaveikai, tikslumui, stabilumui ir panašiai, reikia kurti arba universalią sistemą, kuri komponentus leistų adaptuoti arba kurti specializuotą sistemą tam tikrai sričiai [13].

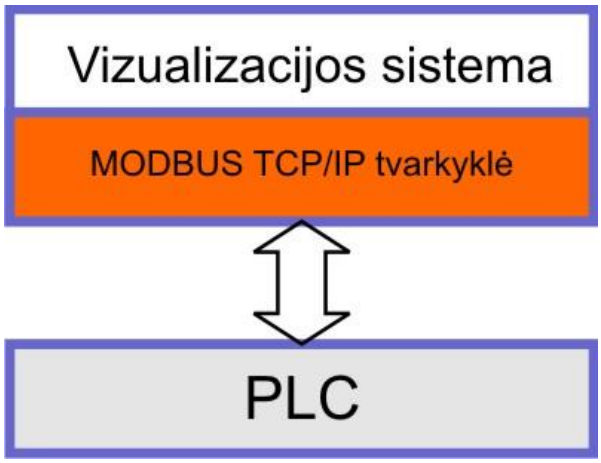
Kuriant vizualizacijos sistemą įvairioms taikymo sritims, iškyla būtinybė ją daryti kuo universalesnę, maksimaliai patikimą ir t.t., kas priveda prie visos eilės problemų sudarant komponentų bibliotekas, realizuojant funkcionalumą. Dėl šių priežasčių vizualizacijos sistema taptų „sunki“, t.y. lėtai veikianti, naudojanti daug resursų, todėl sunkiau įsisavinama ir reiklesnė

aparatūriniam kompiuterio išpildymui [13]. Tačiau tokia sistema tenkintų visų verslo sričių specifiką, būtų labiau konkurentabili.

Vizualizacijos sistema, skirta konkrečiai sričiai yra daug lengviau kuriama, jai lengviau patenkinti specifinius reikalavimus (nes jų yra mažiau nei universaliai, arba juos galima praplėsti kitų reikalavimų sąskaita). Kadangi komponentų biblioteka specifinė, ji tampa labiau išbaigta ir vertinga. Tokia vizualizacijos sistema geriau patenkina specifinės verslo srities poreikius, nei universali, tačiau gali konkuruoti tik vienoje srityje, kas riboja jos panaudojimą.

Duomenų šaltiniai

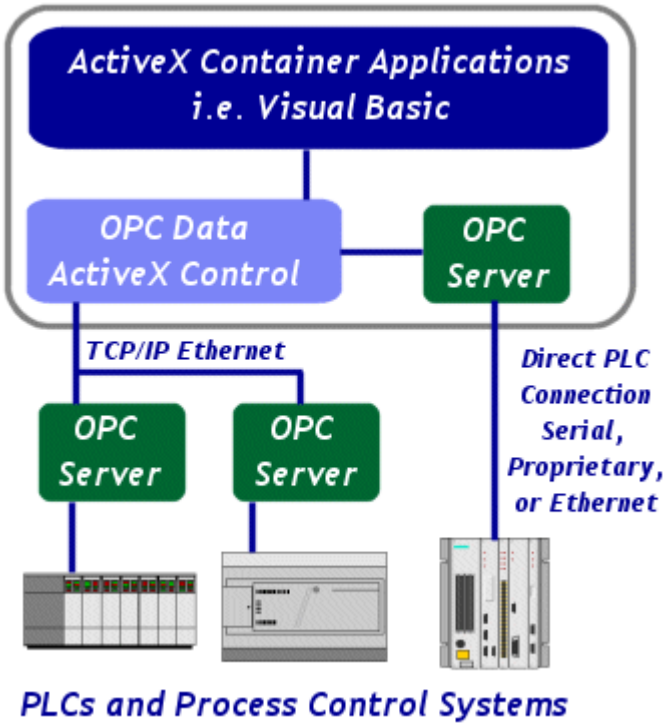
Viena svarbesnių vizualizacijos sistemos savybių yra kokius duomenų šaltinius vizualizacijos sistema palaiko. Šiuo metu labiausiai paplitęs duomenų perdavimo standartas yra OPC (OLE for Proces Control) [14]. Šio standarto pagalba yra supaprastinamas duomenų paėmimas ir perdavimas vizualizacijos sistemoms. Jeigu vizualizacijos sistema palaiko duomenų paėmimą per OPC, ji gali bendrauti su visomis duomenų paėmimo-padavimo sistemomis, kurios realizuotos pagal šį standartą (Pav. 6). Kuriant vizualizacijos sistemą svarbu laikytis kurio nors duomenų perdavimo standarto tam, kad būtų išvengta papildomo darbo kuriant posistemę duomenų paėmimui, apdorojimui ir perdavimui į vizualizacijos sistemą – kitaip tariant dar vieno standarto kūrimo.



Pav. 5 Duomenų paėmimo per TCP/IP tvarkyklę modelis

Tačiau yra procesų, kuriems standartinis OPC duomenų surinkimo ir perdavimo greitis nėra pakankamas. Tokiu atveju tenka duomenų perdavimą optimizuoti. Tam pasiekti yra trys būdai: mažinti perduodamų duomenų kiekį, naudoti optimizuotą OPC [15] arba naudoti tvarkykles tiesioginiam duomenų paėmimui (Pav. 5). Kaip minėta, trimatė vizualizacija yra daugiau orientuojama arba į sudėtingų objektų vizualizavimą, arba į greitų procesų vizualizavimą. Norint

užtikrinti greitą ir kokybišką vizualizaciją, būtini pakankamai dideli duomenų srautai, kas gali būti gan problematiška naudojant OPC technologiją.



Pav. 6 Duomenų mainų pavyzdys per OPC serverį [18]

Sukurta vizualizacijos sistema taip pat gali būti paprastai pritaikyta duomenų paėmimui per OPC sąsają. Tačiau tam, kad būtų išvengta papildomų pastangų kuriant sąsają duomenų paėmimui per OPC serverį, buvo realizuota MODBUS TCP tvaklyklė, leidžianti pasiimti duomenis tiesiai iš įranginio be jokių tarpininkų. Tai ne tik supaprastino sistemos duomenų mainų kūrimo dalį, tačiau ir ženkliai paspartino pačius duomenų mainus.

Pasirinktas sprendimas

Toliau bus apibūdinami pasirinkti technologiniai, architektūriniai bei metodologiniai sprendimai ir juos lėmusios priežastys.

Naudojamos technologijos

Sprendimas naudojamoms technologijoms pasirinktas atsižvelgiant į daugelį veiksnių. Pagrindinis faktorius, apsprendęs praktiškai visų likusių technologijų pasirinkimą buvo trimatės

grafikos variklis. Kaip minėjau trimatės grafikos kūrimo priemonių apžvalgoje, šiuo metu plačiausiai naudojamos dvi technologijos: atvirojo kodo sistema OpenGL ir kompanijos Microsoft produktas DirectX. Microsoft DirectX technologiją pasirinkau todėl, kad ją galima naudoti koduojant C# kalba. Kadangi ši kalba man geriausiai žinoma, šio projekto praktinė dalis ir buvo realizuota C#. Naudojant C# ir DirectX technologijas vienintelis operacinės sistemos pasirinkimas buvo Microsoft Windows. Kadangi šis projektas orientuotas ne tiek į konkrečių technologijų panaudojimą kiek į trimačių elementų panaudojimą vizualizacijos sistemose nėra tiek svarbu kurių technologijų pagrindu jis buvo realizuotas.

Architektūrinis sprendimas

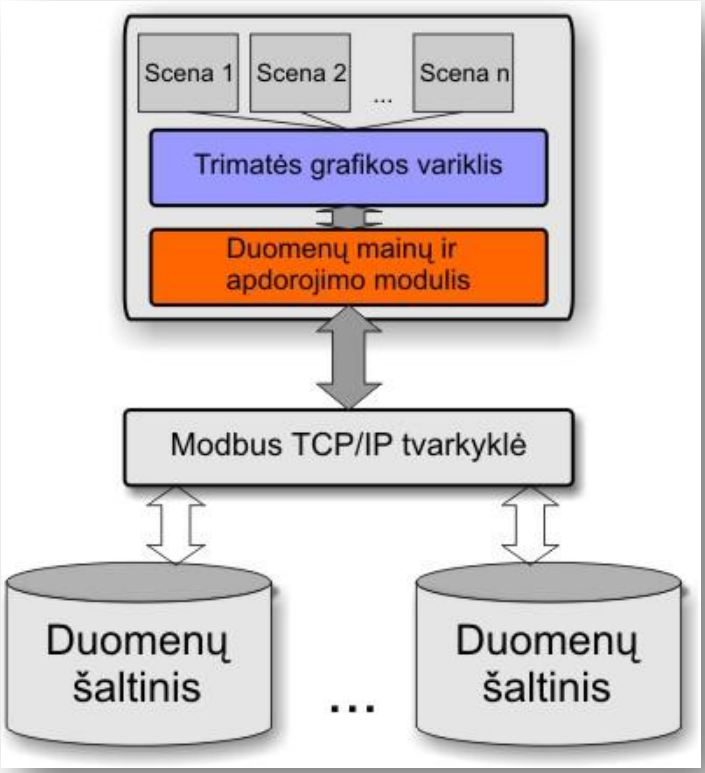
Kitas svarbus realizuoto projekto momentas – duomenų mainai. Kaip minėjau skyriuje „Duomenų šaltiniai“ buvo sukurtos tvarkyklės, leidžiančios mainytis duomenimis tiesiogiai, be jokių tarpininkų. Šiuo metu realizuota vizualizacijos sistema veikia tik su įrenginiais (paprastai programuojamaisiais loginiais valdikliais) kurie palaiko Modbus TCP/IP arba Modbus RTU protokolus. Realiai veikiantis objektas, kuriame bus išbandyta sukurta vizualizacijos sistema, yra valdomas įrenginių, kurie palaiko Modbus TCP protokolą, todėl ir buvo realizuotos būtent šio protokolo tvarkyklės. Objekte duomenys surenkami per OPC serverį, todėl buvo galima realizuoti ir duomenų mainus per jau egzistuojančius OPC, tačiau to buvo atsisakyta dėl toliau išvardintų kelių priežasčių.

OPC technologija paremta ActiveX. Tai reiškia, jog tai yra pasenusi technologija. Naujoji OPC versija OPC DA (**O**LE for **P**roces **C**ontrol. **D**ata **A**cces) kurios pagrindas nebėra ActiveX technologijos, o realizuota naudojant Microsoft .NET platformą, yra šviežiai pasirodžiusi ir kolkas plačiai nėra paplitusi. Naujai kuriamam produktui naudoti pasenusias technologijas nėra tikslinga, jei nėra specialių reikalavimų ar tam tikrų apribojimų.

Norint naudotis OPC serverio duomenimis, jame reikia sukurti duomenų žymes (*angl. data tags*). O tai daryti jau realizuotame ir veikiančiame objekte nėra rekomenduotina. Sukūrus papildomas žymes, nereikalingai padidinamas skaitomų/rašomų duomenų srautas, o papildomos tiek tinklo, tiek ir įrangos apkrovos nėra teigiamas dalykas.

Tvarkyklės taip pat papildomai apkrauna duomenis teikiančią įrangą, tačiau yra apeinamas OPC serveris. Be to, įranga apkraunama tik startavus vizualizacijos sistemą. Kadangi sistema paleidžiama tik testavimo ir naudojimo bandymo metu, papildomos apkrovos laikas sumažinamas minimaliai. Tačiau pats svarbiausias momentas, dėl ko buvo realizuotos tvarkyklės yra tas, jog sukurtos sistemos naudojimas naudojant tvarkykles nesukelia jokių pakeitimų egzistuojančioje

sistemoje. Įvertinus čia išvardintus faktus bei apribojimus buvo pasirinktas daugiasluoksnis sistemos modelis (Pav. 7).



Pav. 7 Sistemos architektūros modelis

Naujai iškilusios problemos

Šiame skyriuje aptariamos naujai iškilusios problemos projekto realizavimo metu ir jų sprendimas. Taip pat aptariami galimi kiti problemų sprendimo variantai.

Aparatūrinių resursų ribotumas

Visa programinė įranga, naudojanti trimatę grafiką tampa pakankamai priklausoma nuo aparatūrinių resursų. Realizuojant projektą, buvo pasiektas savotiškas turimų aparatūrinių resursų limitas – vizualizacijos scenos atnaujinimo dažnis buvo nukritęs iki 8 kadrų per sekundę. Tai reiškia, kad toks atnaujinimo periodiškumas nėra pakankamas norint užtikrinti efektyvų ir gražų vizualizacijos vaizdą, pateikiamą vartotojui. Minimalus atnaujinimo periodiškumas neturėtų būti mažesnis nei 12 kadrų per sekundę.

Didelę įtaką veikimo spartai turi scenos detalumas. Vienas iš būdų suefektyvinti atnaujinimo periodiškumą yra sumažinti scenos detalumą – kitaip tariant atvaizduojamų poligonų skaičių. Šis metodas ir buvo panaudotas tam, kad būtų pasiektas 16 kadrų per sekundę atnaujinimo dažnis.

Bendrąja prasme trimačių objektų optimizacija yra pakankamai sudėtingas uždavinys [19]. Yra įvairių metodų – pradedant poligonų skaičiaus mažinimu taikant įvairius apriksimacijos algoritmus, baigiant apšvietimo, atspindžių, šešėlių ir t.t. optimizacijomis.

Naudojimo sunkumai

Lyginant su klasikine (dvimate) vizualizacijos sistema, trimatės vizualizacijos naudojimas yra sudėtingesnis. Pagrindiniai skirtumai, kurie apsunkino vizualizacijos su trimačiais objektais naudojimą:

- **Judantys objektai** – atsiradus objektams, kurie keičia padėtį erdvėje, tapo sudėtingiau orientuotis scenoje ir pateikiamuose duomenyse. Net ir besisukatis apie savo ašį objektas įneša „sutrikimo“ į sceną – taip yra todėl, kad su besisukančiu trimačiu elementu kartu sukasi ir jį atitinkantys duomenys.
- **Padidėjęs lygmenų skaičius** – naudojant trimačius elementus kartais tampa labai sunku arba netikslinga naudoti tik du lygmenis duomenims pateikti. Dvimatėje erdvėje praktiškai visuomet (su nedidelėmis išimtimis) užtenka dviejų lygmenų[3] duomenims pateikti. Yra teigiama, jog kuo mažesnis lygmenų (bet kokios hierarchijos) skaičius, tuo naudotojas ją geriau įsimena – tokiu būdu supaprastinams sistemos naudojimas.

Sistemos stabilumas

Žaidžiantis trimačius kompiuterinius žaidimus (ypač sudėtingesnius) skaitytojas žino, jog trimatė grafika linkusi „kibti“. Trimatės grafikos perteikimas (*angl. rendering*) yra sudėtingas aparatūrinis procesas, apdorojantis milžiniškus kiekius grafinės informacijos. Net ir naudojantis šiuolaikinėmis technologijomis vis dar pakankamai sunku užtikrinti grafikos darbo stabilumą. Kaip minėta, vizualizacijos sistema gali būti naudojama labai sudėtingiems procesams vizualizuoti, todėl sistemos stabilumas ir nepertraukiamas darbas yra vienas esminių akcentų, kuriuos reikia prisiminti projektuojant vizualizacijos sistemas.

Sukurtą sistemą buvo bandoma padaryti pakankamai atsparią „kibimams“. Tačiau negaliu teigti, kad sukurtos sistemos stabilumas yra užtikrintas. Norint užtikrinti sistemos stabilumą

[3] Lygmuo (duomenų lygmuo) - čia ir toliau tekste duomenų lygmeniu vadinama scenoje esantis ekrano elementų (iššokančių langų, meniu ir pan. elementų) vienu metu esančių ir uždengiančių vienas kitą ekrane, skaičius.

pakankamai aukštame lygmenyje, būtini sistemos testai naudojant kiek įmanoma įvairesnę techninę įrangą. Tik tokiu būdu testuojant būtų galima teikti, kad sistemos stabilumas yra tam tikro lygmens naudojant nurodytą aparatūrą.

Reikalavimai sistemai

Toliau yra išverdinti pagrindiniai reikalavimai, iškelti sukurtai sistemai.

Galimi sistemos naudotojai

Sukurtos sistemos potencialius naudotojus galima suskirstyti į tris grupes:

- Pramonės ar kitų specializuotų šakų vizualizuojamų ir valdomų procesų operatoriai
- Individualių namų savininkai, įsirengę namų automatiką („protingo namo sistemą“)
- Biurų pastatų, įvairių didesnių kontorų savininkai/operatoriai, kuriems aktualus sistemos stebėjimas

Bendrai apibūdinant sistemos naudotojus galima teigti, jog šios sistemos naudotojų (išskyrus nedideles išimtis) patirtis IT srityje yra arba labai nedidelė, arba specifinė (tik tam tikrų taikomųjų programų žinojimas), kas šiek tiek apsunkina užduotį tuo atžvilgiu, jog sistemos naudotojas gali turėti papildomų problemų perprantant šios sistemos teikiamas galimybes.

Pramonės ar kitų specializuotų šakų vizualizuojamų ir valdomų procesų operatoriai

- Tai objekto, kuriame įdiegta ši sistema, operatorius
- Naudotojo sprendžiami uždaviniai:
    - Objekto būsenos stebėjimas realiame laike
    - Operatyvi avarinių situacijų analizė ir šalinimas
    - Detalus atskirų objekto dalių stebėjimas tam tikriems poreikiams tenkinti
    - Parametrų uždavimas
- Tai naudotojas, kurio patirtis dalykinėje srityje yra pakankamai didelė, tačiau IT išmanymas žemas. Tai gali būti tiek eilinis operatorius, tiek objekto meistras ar net įmonės vadovas.
- Tai asmuo, neturintis nei fizinių, nei protinių negalių, ar kitokių tam tikrų apribojimų, kurie turėtų specifinės įtakos kuriamai vizualizacijos sistemai. Visus vartotojo poreikius turėtų tenkinti standartinės darbo aplinkos adaptavimo funkcijos (šrifto dydis, spalva, ...)

Individualių namų savininkai, įsirengę namų automatiką („protingo namo“ sistemą)

- Tai namo, kuriame įrengta „protingo namo“ sistema, gyventojai

- Naudotojo sprendžiami uždaviniai:
    - Namo paramentrų stebėjimas dabartiniu momentu
    - Gedimų nustatymas
    - Parametrų (kambarių temperatūros, apšvietimo, ...) nustatymas
- Paprastai tai eilinis naudotojas (ne automatikos specialistas), kurio IT išmanymas gali būti ir labai didelis, ir labai mažas.
- Šis asmuo gali turėti fizinių negalių:
    - Silpna rega
    - Silpna klausa
    - Ir t.t.

      Bene vienintelis elementas, apribojantis naudotojo galimybes, yra jo rega. Todėl kuriant galutinę šios sistemos versiją būtina atsižvelgti į galimybę kaip įmanoma lanksčiau išspresti silpnos regos problemą. Dabar sukurtai sistemai papildomi tokio pobūdžio reikalavimai nebus keliami.

<u>Biurų pastatų, įvairių didesnių kontorų savininkai/operatoriai, kuriems aktualus sistemos stebėjimas</u>

Šios naudotojų kategorijos keliami reikalavimai pilnai padengiami prieš tai aprašytų naudotojų kategorijų, todėl atskirai nagrinėjami nebus.

Naudotojų prioritetai:

Pagal svarbą, naudotojus reikėtų rūšiuoti taip (prioriteto mažėjimo tvarka) :

- **Pramonės ar kitų specializuotų šakų vizualizuojamų ir valdomų procesų operatoriai** – šie naudotojai svarbiausi todėl, kad šios srities operatorių darbas pats atsakingiausias, kur klaidos kaina pati didžiausia.
- **Individualių namų savininkai** – ši naudotojų kategorija svarbi tuo, kad jiems ypač aktualu, jog sistema ne tik patikimai funkcionuotų, bet ir atrodytų kuo įspūdingiau (prestižo, meniniais ar kitais sumetimais).
- **pastatų, įvairių didesnių kontorų savininkai/operatoriai**

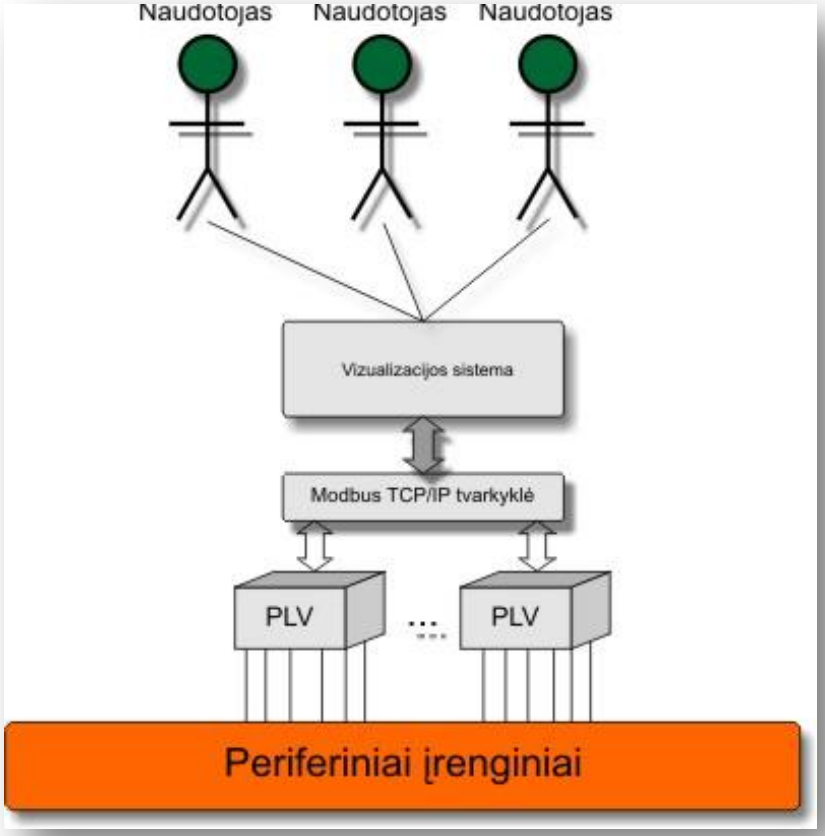
Šitoks naudotojų grupių prioretizavimas yra sąlyginis. Kadangi šio produkto nenaudos atsitiktiniai asmenys, o tik tam tikri, su panašiais rekalavimais, naudotojai, todėl ir prioretizavimas netenka didelės prasmės, nes į visus visų naudotojų keliamus reikalavimus būtina atsižvelgti.

Sistemos diegimo ir testavimo aplinka

Kadangi sistema skirta atvaizduoti fiziniams procesams, ji glaudžiai bendradarbiauja su automatikos sistema, įdiegta objekte. Imami ir atvaizduojami duomenys iš temperatūros, traukos, slėgio, apšviestumo ir t.t. daviklių, taip pat iš įvairių įrenginių, tokių kaip programuojamieji loginiai valdikliai (PLV), dažnio keitikliai, pavaros, trieigiai vožtuvai, sklendės ir t.t. Tai paprastai centralizuota sistema, kur procesą valdo PLV, prie kurio yra prijungti visi išvardinti elementai. Duomenys apie procesą yra imami iš PLV, o užduotys (parametrai procesui) yra taip pat nurodomi PLV. Tam, kad būtų supaprastintas duomenų paėmimas ir įrašymas į PLV, komunikacijos bus vykdomos per realizuotą Modbus TCP protokolo tvarkyklę. Galimas ir decentralizuotas proceso valdymas su keliais PLV, tokiu atveju duomenys imami iš visų reikalingų taškų. Toks decentralizavimo atvejis, kuomet nėra centrinio PLV o kiekvienas automatizuotos sistemos elementas turi savo individualų procesorių (tokia sistema ypač paplitus „protingo namo“ instaliacijose) kolkas nebus atskirai detaliai nagrinėjamas, tačiau jis iš princpo nesiskiria nuo klasikinės centralizuotos automatizavimo sistemos – dėl šios priežasties sukurta sistema gali būti pritaikyta ir tokioms sistemoms. Pagrindinis skirtumas toks, jog duomenys imami ne iš centrinio (ar kelių) PLV, bet iš kiekvieno įrenginio individualiai per adapterį, prijungtą prie duomenų magistralės.

Sistemos veiklos kontekstas

Šiame skyriuje trumpai apibūdinsiu sistemos veiklos kontekstą (Pav. 8) kuriame numatyta dirbti sukurtai sistemai.



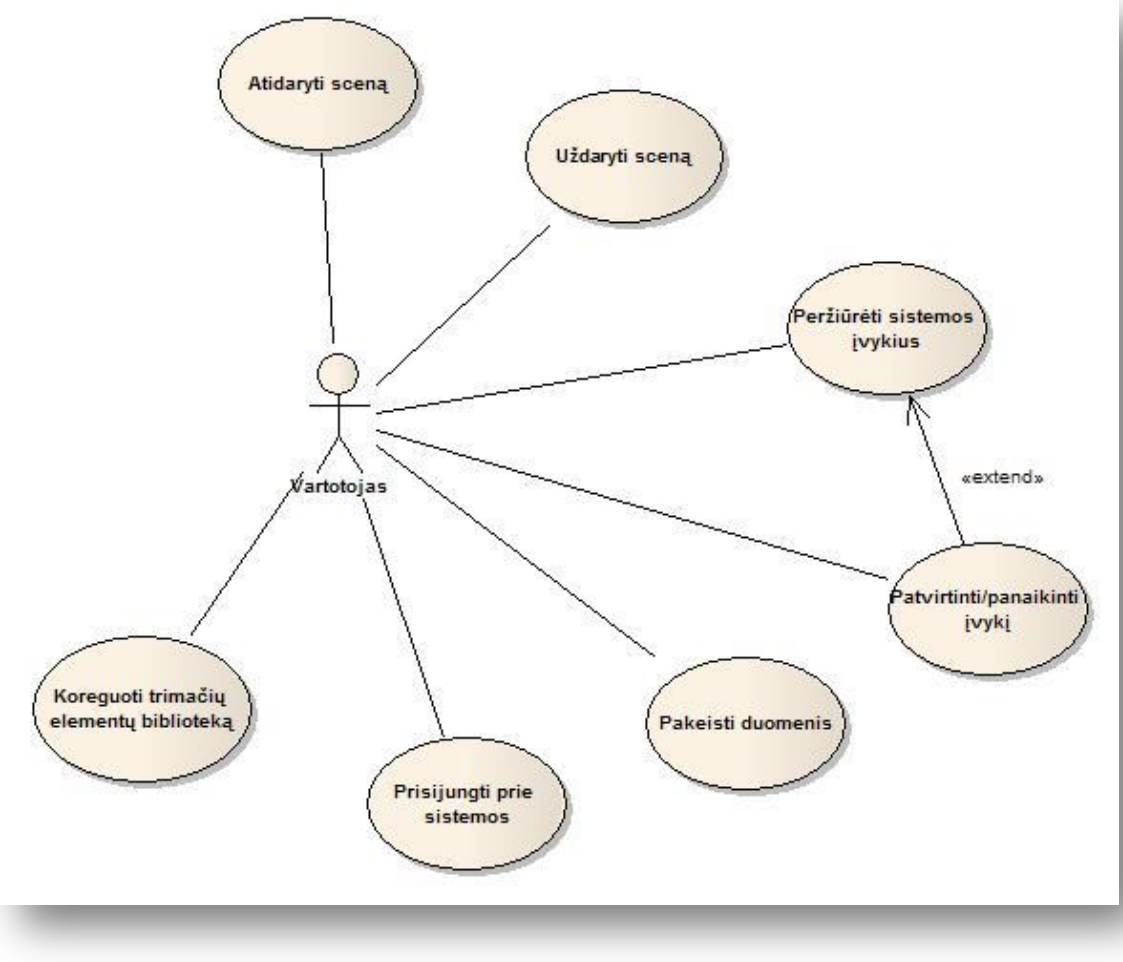
Pav. 8 Sistemos veiklos kontekstas

Vizualizacijos sistemos aplinką galima apibūdinti taip:

- Vizualizacijos sistemos tecninis kontekstas
  - Įvairūs duomenų mainų protokolo keitikliai, adapteriai
  - PLV, paskirstyta sistema (DCS) arba kiti valdantieji centralizuoti ar decentralizuoti įrengimai
  - Visa objekte esanti kontroliuojančių įrenginių periferija: siurbliai, orapūtės, dūmtraukės, pavaros, sklendės, slėgio/CO_2 ir kiti davikliai ir t.t.
  - Kompiuteriniai, duomenų mainų, kiti komunikaciniai tinklai
- Vizualizacijos sistemos programinis kontekstas
  - Duomenų mainų Modbus TCP/IP tvarkyklės
  - Protokolo keitiklių ar adapterių tvarkyklės
  - Kita tame pačiame tinkle egzistuojanti ir veikianti programinė įranga, su kuria įmanomi tam tikri duomenų mainai
- Žmogiškasis kontekstas
  - Operatorius – pastoviai dirbantis su sistema žmogus ar žmonių grupė

Sistemos ribos

Vizualizacijos sistemos panaudos atvejų modelis pateiktas Pav. 9.



Pav. 9 Sistemos ribos. Panaudos atvejų modelis

Taip pat pateikiamas panaudos atvejų detalus aprašas.

lentelė 1. Panaudos atvejo ,,uždaryti sceną" aprašas

1. Uždaryti sceną	
Aktorius	Operatorius
Aprašas	Aktyvi scena uždaroma
Prieš-sąlyga	Turi būti atidaryta nors viena scena
Sužadinimo sąlyga	Reikia peržiūrėti kitus duomenis, arba scena nebereikalinga
Po-sąlyga	-

lentelė 2. Panaudos atvejo ,,atidaryti sceną” aprašas

2. Atidaryti sceną	
Aktorius	Operatorius
Aprašas	Atidaroma pasirinkta scena
Prieš-sąlyga	Scena turi būti sukurta
Sužadinimo sąlyga	Operatorius ar kūrėjas atidaro sceną iš vartotojo sąsajos
Po-sąlyga	Scena atidaroma

lentelė 3. Panaudos atvejo ,,peržiūrėti sistemos įvykius” aprašas

3. Peržiūrėti sistemos įvykius	
Aktorius	Operatorius
Aprašas	Operatorius peržiūri avarijų sąrašą norėdamas išsiaiškinti ir lokalizuoti gedimus
Prieš-sąlyga	Turi būti sukurtas avarijų sąrašas
Sužadinimo sąlyga	Operatorius, pasinaudodamas vartotojo sąsaja iškviečia aktyvių avarijų sąrašą
Po-sąlyga	Parodomas aktyvių avarijų sąrašas

lentelė 4. Panaudos atvejo ,,patvirtinti/panaikinti įvykį" aprašas

4. Patvirtinti/panaikinti įvykį	
Aktorius	Operatorius
Aprašas	Panaikinamas arba patvirtinamas sistemos įvykis tam, kad pašalinus įvykį sukėlusias priežastis jis būtų pašalinamas iš įvykių sąrašo
Prieš-sąlyga	Turi būti įvykęs nors vienas įvykis
Sužadinimo sąlyga	Kūrėjas pasinaudodamas varototjo sąsaja iškviečia įykių sąrašą ir pasirinkus norimą įvykį spaudžiamas mygtukas „patvirtinti“
Po-sąlyga	Įvykis išnyksta iš įvykių sąrašo arba atsiranda indikacija apie įvykio patvirtinimą

lentelė 5. Panaudos atvejo ,,prisijungti prie sistemos" aprašas

5. Prisijungti prie sistemos	
Aktorius	Operatorius
Aprašas	Operatorius prisijungia prie sistemos norėdamas pradėti darbą su ja
Prieš-sąlyga	Operatorius turi turėti sukurtus prisijungimo duomenis
Sužadinimo sąlyga	Operatorius pasinaudodamas vartotojo sąsaja prisijungia prie sistemos
Po-sąlyga	-

lentelė 6. Panaudos atvejo ,,peržiūrėti sistemos įvykius" aprašas

6. Peržiūrėti sistemos įvykius	
Aktorius	Operatorius
Aprašas	Parodomi sistemoje esantys įvykiai
Prieš-sąlyga	-
Sužadinimo sąlyga	Operatorius pasinaudodamas vartotojo sąsaja iškviečia įvykių sąrašą
Po-sąlyga	-

Sistemos architektūros specifikacija

Dokumente pateikiama sukurtos vizualizacijos sistemos architektūros apžvalga. Pateikiama veiklos, sekų, būsenų, klasių diagramos, kurios ir aprašo sukurtos sistemos architektūrą. Šiame dokumente taip pat galima rasti sistemos darbo aplinką bei trumpą aprašą kaip ir su kokia įranga sistema dirbs.

Architektūros tikslai ir apribojimai

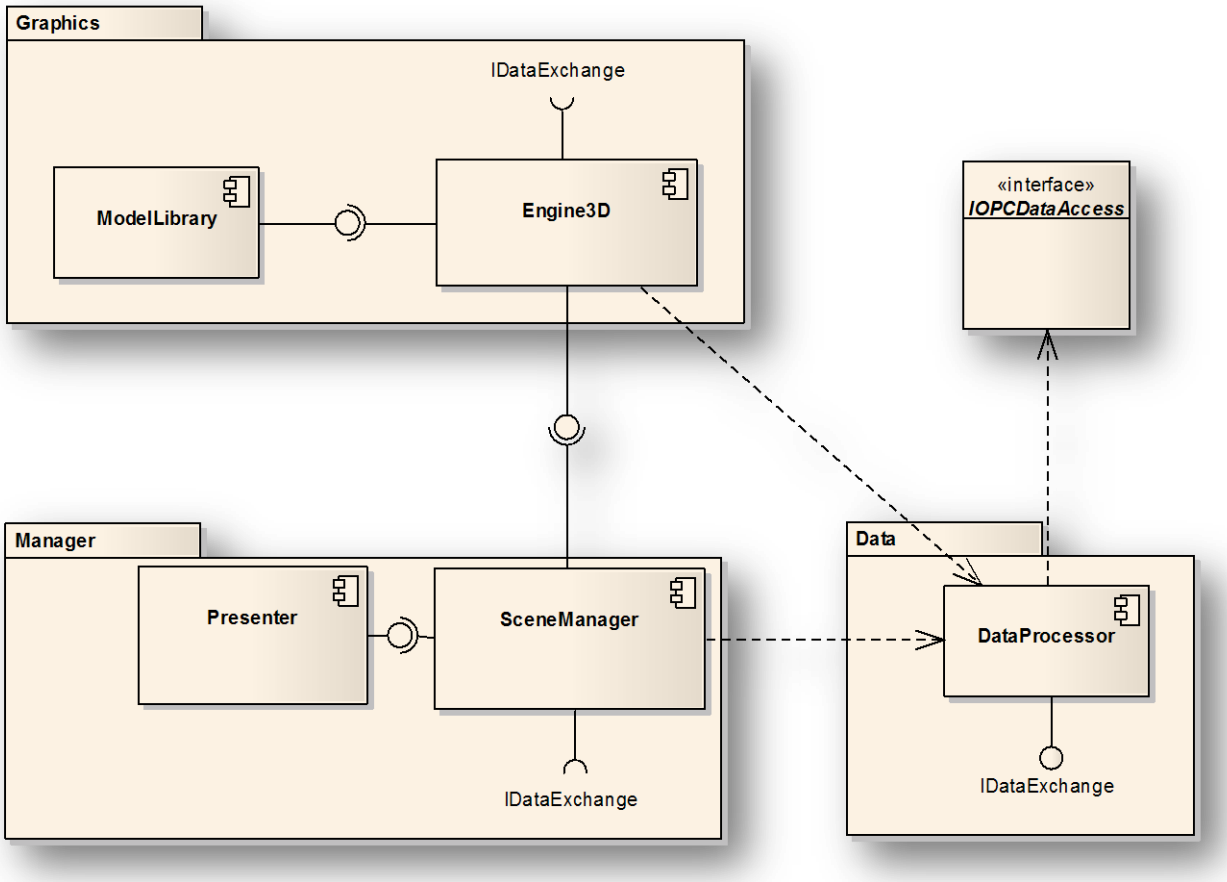
Kadangi sistema turi dirbti su įvairių gamintojų technine įranga (ABB, Schneider-electric, Siemes, Beckhoff ir t.t.), reikia duomenų mainų sąsajos tarp vizualizacijos sistemos ir techninės įrangos (PLV). Tam buvo realizuota Modbus TCP/IP tvarkyklė. Ji pateikia sąsają duomenų mainams tarp vizualizacijos sistemos ir procesą valdančių ar duomenis atiduodančių įrenginių. Tai standartinis sprendimas, naudojamas duomenų mainų užtikrinimui.

Sistemos statinis vaizdas

Visa sistema suskaidyta į tokius tris atskirus paketus:

- **Graphics** – tai sistemos dalis, kuri skirta darbui su grafiniais elementais. Pradedant scenų apipavidalinimu, baigiant trimačių objektų įkėlimu į biblioteką ar sceną.
- **Manager** – tai sistemos dalis, kuri atsakinga už visą sistemos logiką. Tai būtų scenų kūrimas, šalinimas, darbo organizavimas, tarpininko tarp kitų paketų vaidmuo ir t.t.
- **Data –** tai sistemos dalis, skirta duomenų mainams tarp Modbus TCP/IP tvarkyklės ir kuriamos vizualizacijos sistemos. Taip pat ji skirta ir duomenų paruošimui (tam tikro tipo duomenų, pvz. String, Float, Integer) tam, kad vaizdavimui jie būtų pateikti jau pilnai suformuoti (transformuoti, nufiltruoti ir pan.)

Pav. 10 pateiktas sistemos išskaidymas į paketus.



Pav. 10. Sistemos suskaidymas į paketus

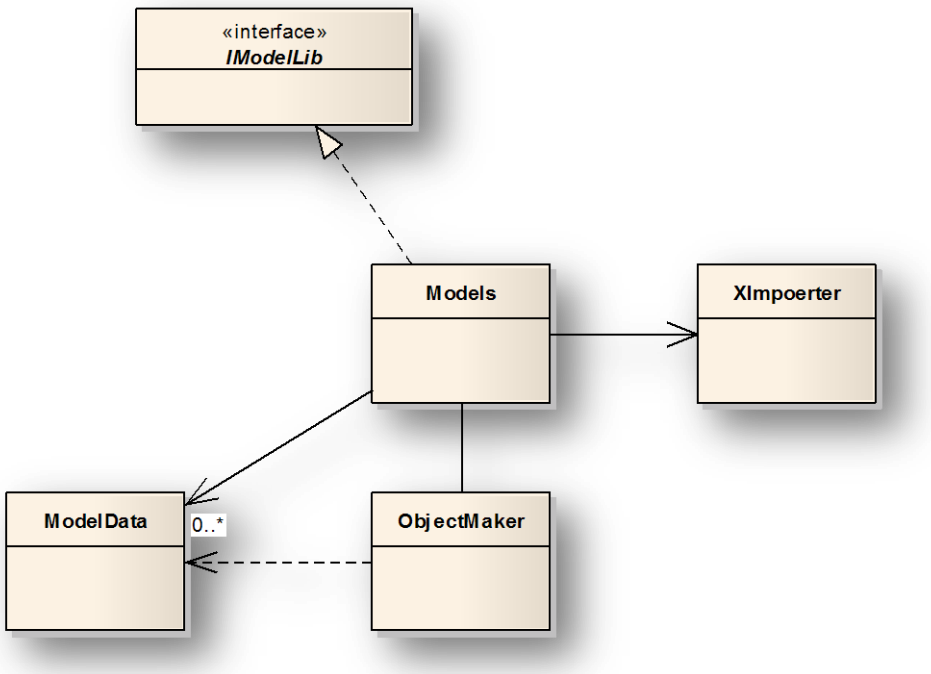
Paketas „Graphics“

Šis paketas sudarytas iš dviejų komponentų:

- ModelLibrary
- Engine3D

Komponentas „ModelLibrary“ skirtas trimačių objektų bibliotekos funkcionalumui realizuoti. Jis atsakingas už tokius veiskmus:

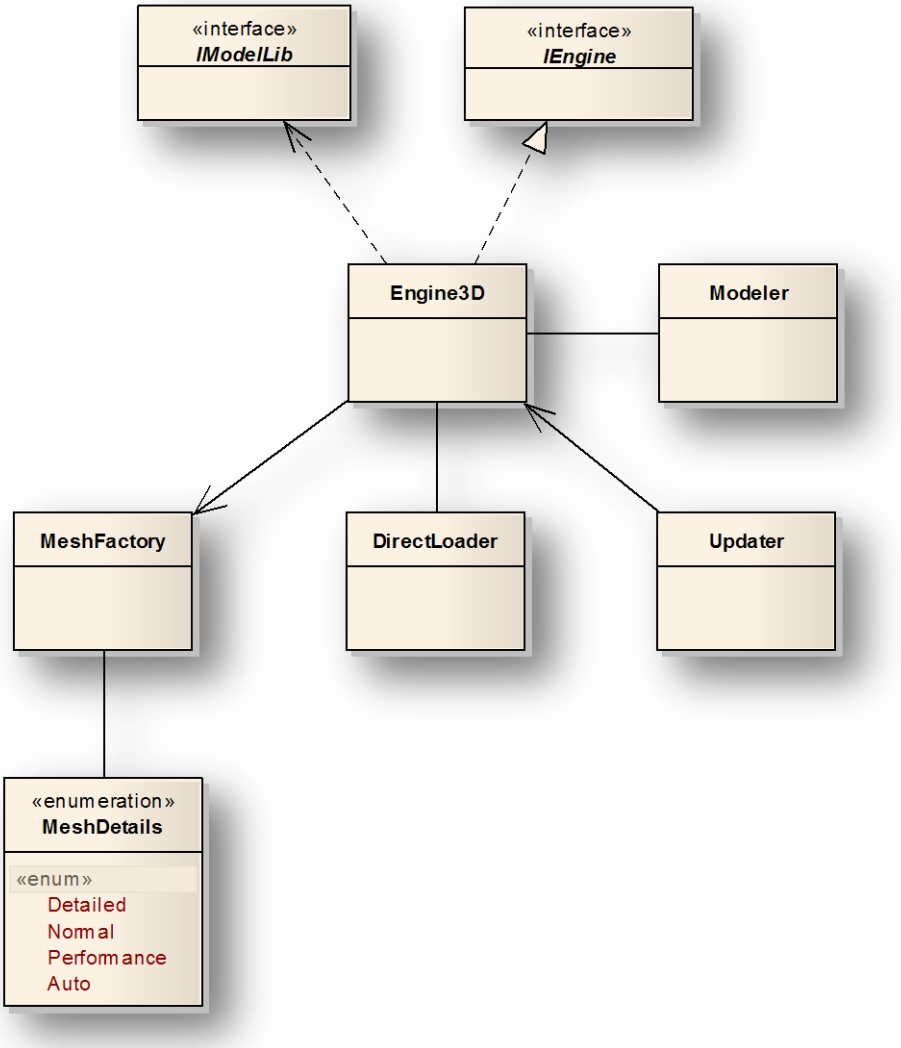
- Importuoti modelį iš failo, kurio formatas *.x
- Išsaugoti importuotą modelį savo vidinėje modelių saugykloje (bibliotekoje)
- Pateikti bibliotekoje egzistuojančių modelių sąrašą
- Suformuoti visus reikiamus duomenis, įkelti juos į atmintį ir grąžinti suformuotą (mesh) objektą (modelį) kuris gali būti keliamas į sceną
- Ištrinti nurodytą modelį



Pav. 11. Komponento "ModelLibrary" klasių diagrama

Komponentas „Engine3D“ skirtas dvimačių ir trimačių objektų vaizdavimui scenoje. Jis atsakingas už šiuos veiksmus:

- Suformuoti bendrą scenos vaizdą iš dvimačių ir trimačių elementų
- Gavus naujus duomenis iš sistemos, atnaujinti sceną
- Manipuliuoti objektais ekrane: scenų langais, meniu ir pan.



Pav. 12. Komponento "Engine3D" klasių diagrama

Paketas „Manager"

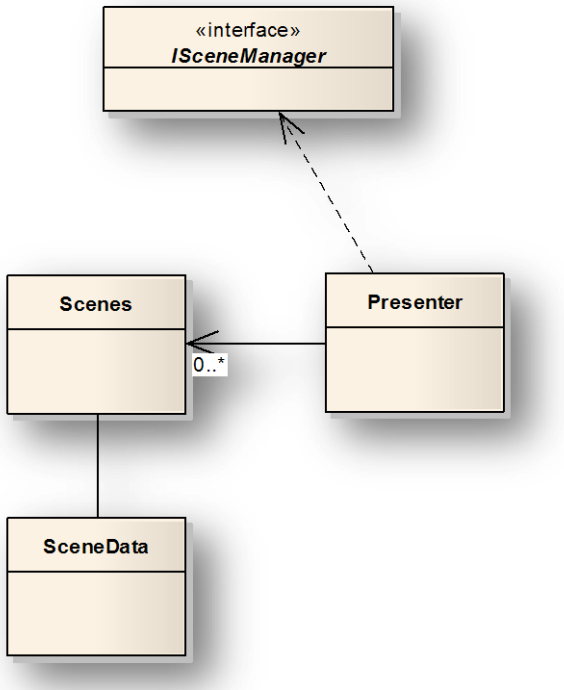
Paketas „Manager" sudarytas iš dviejų komponentų:

- Presenter
- SceneManager

Komponentas „Presenter" skirtas logikos valdymui, kuomet reikia rodyti objektus ekrane. Jis valdo komponentą „Engine3D" jam formuodamas užduotis. Jis apsprendžia kurioje vietoje ir kaip atrodanti scena, meniu ar kitas vartotojo sąsajos elementas turi atsirasti ekrane. Šis komponentas atsakingas už šiuos veiksmus:

- Saugoti duomenis apie sceną (vaizdas, pozicija, tam tikri vartotojo požymiai, elementai scenoje)

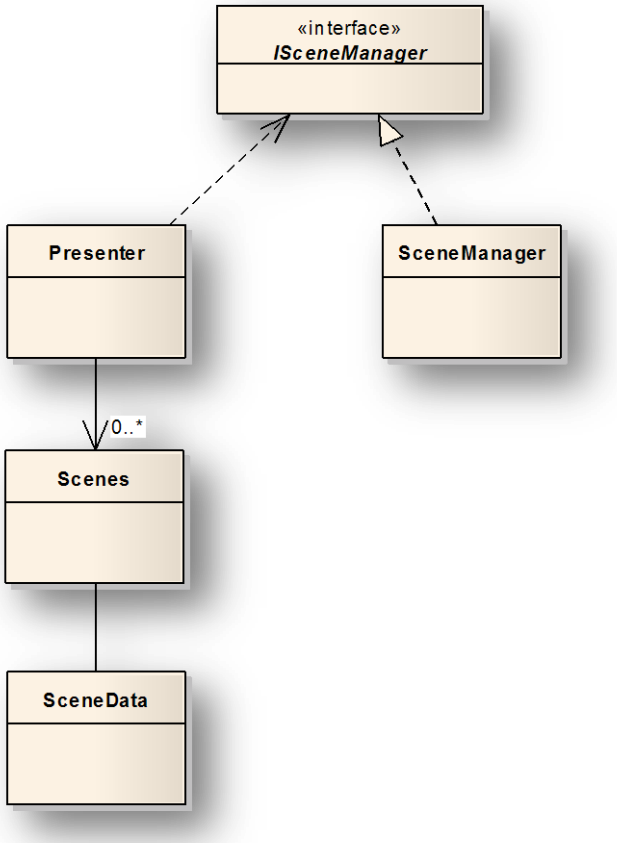
- Formuoti užduotis komponentui „Engine3D“: nurodyti kokius objektus ir kur vaizduoti



Pav. 13. Komponento "Presenter" klasių diagrama

Komponentas „SceneManager“ skirtas scenų valdymui. Atsakingas už šiuos veiksmus:

- Saugoti ir pateikti sistemoje sukurtų scenų sąrašą
- Leisti atlikti manipuliacinius veiksmus su scenomi: kurti, trinti, keisti, atidaryti, uždaryti, eksportuoti ir importuoti

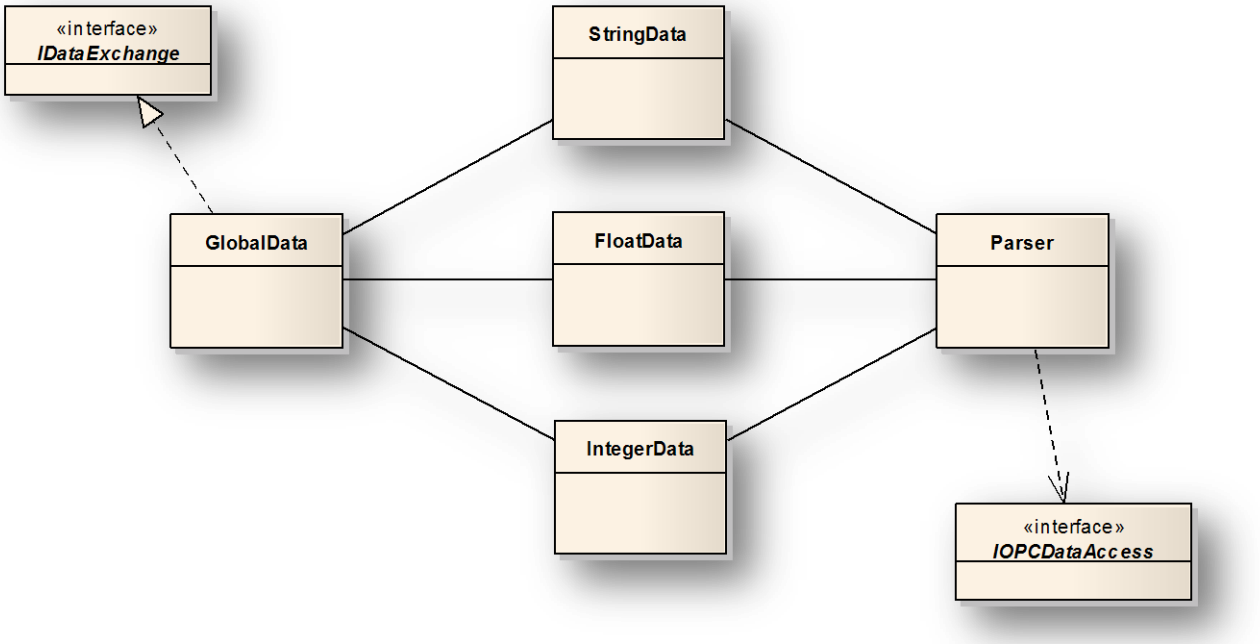


Pav. 14. Komponento "SceneManager" klasių diagrama

Paketas „Data“

Paketą „Data“ sudaro vienintelis komponentas „DataProcessor“. Jo pagalba keičiamasi duomenimis tarp vizualizacijos sistemos ir OPC serverio. OPC serveris savo ruožtu užtikrina duomenų mainus tarp „DataProcessor“ komponento ir techninės įrangos, veikiančios objekte. Šis komponentas atsakingas už šiuos veiksmus:

- Duomenų žymių (tags) sąrašo saugojimas sistemoje
- Leisti atlikti manipuliacinius veiksmus su žymių sąrašu: kurti, trinti, keisti žymes.

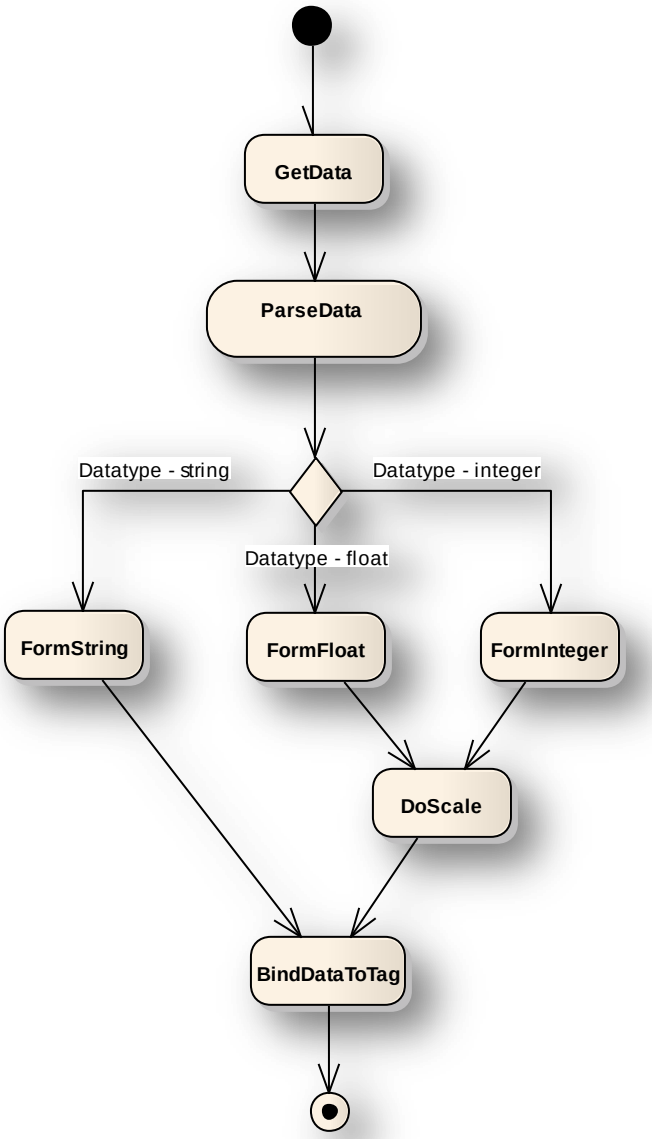


Pav. 15. Komponento "DataProcessor" klasių diagrama

Sistemos dinaminis vaizdas

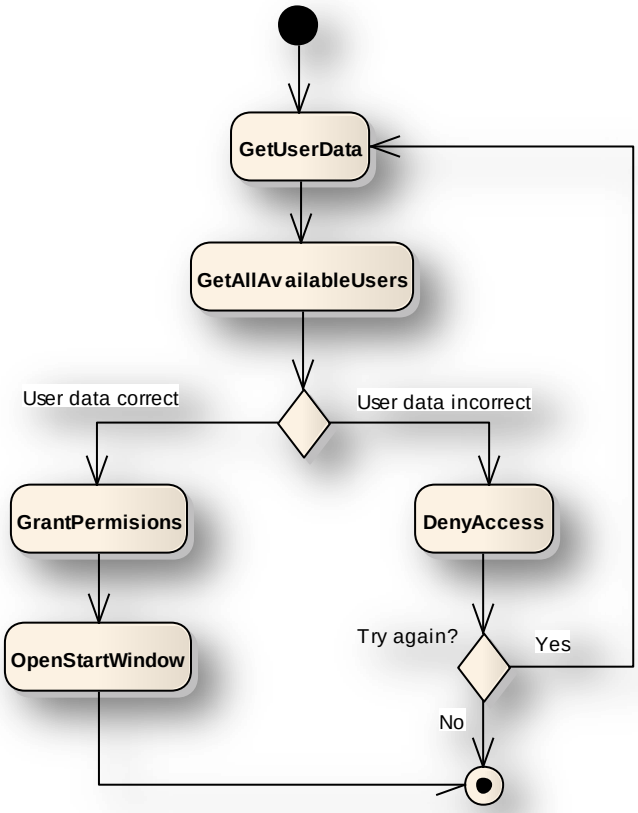
Šiame skyriuje pateikiamas sistemos dinaminis vaizdas – veiklos ir būsenų diagramos, skirtos atvaizduoti sistemos darbą.

Veiklos diagrama duomenų iš paėmimui ir pririšimui prie žymės



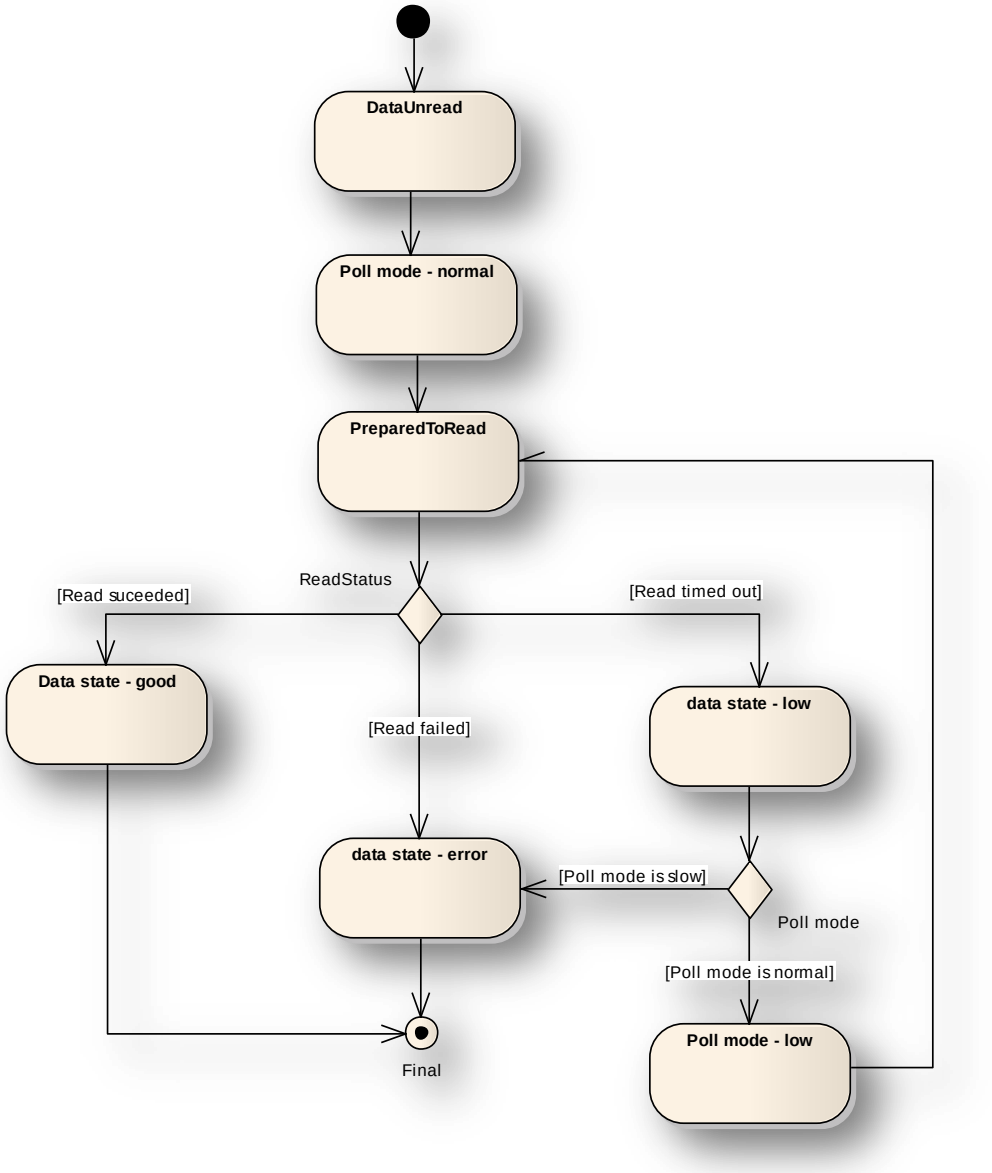
Pav. 16 Duomenų pririžimo prie žymės veiklos diagrama

Veiklos diagrama prisijungimui prie sistemos



Pav. 17 Prisijungimo prie sistemos veiklos diagrama

Būsenos diagrama duomenų kokybės kitimui atvaizduoti



Pav. 18 Duomenų būsenos diagrama

Sistemos kokybė

Šiame skyriuje pateikiamas sistemos kokybės vertinimas šiais požiūriais:

- Išplečiamumas
- Pernešamumas
- Stabilumas

Išplečiamumas

Kadangi sistema išskaidyta į tarpusavyje per sąsajas bendraujančius komponentus, supaprastėja atskirų jos dalių taisymo ar tobulinimo klausimas. Tol, kol sąsaja (interface) lieka nepakitusi, bet kurie pakeitimai komponento viduje yra laisvai galimi ir nereikalauja kitų komponentų sąsajų ar funkcionalumo keitimo. Realizavus naują komponentą su tam tikru papildomu ar specifiniu funkcionalumu, jis lygiai taip pat kaip ir kiti komponentai bendrautų per tą pačią (ar su atitinkamais pakeitimais jei tai būtina) sąsają.

Pernešamumas

Sistemos pernešamumas tarp skirtingų platformų negalimas, todėl kad jos kūrimas paremtas vien tik Microsoft technologijomis. Jeigu būtų buvęs pasirinktas kitas rimatės grafikos variklis, pvz. OpenGL, būtų įmanomas sistemos migravimas tarp platformų. Tačiau to buvo atsisakyta, nes tai labai padidintų sistemos sudėtingumą ir pasunkintų palaikomumą, ko pirmiausia ir buvo bandoma išvengti.

Stabilumas

Viena iš priežasčių, kodėl sistema kuriama Microsoft technologijų pagrindu yra ta, kad naudojant vieną technologijų liniją supaprastėja sistemos integravimo uždavinys, kūrimas ir palaikymas. To pasekoje paprasčiau užtikrinti sistemos stabilumą.

Eksperimentas

Kaip minėta, sukurta vizualizacijos sistema yra realiai veikiančios dvimatės vizualizacijos sistemos analogas, panaudojantis trimačius elementus. Ši sistema yra planuojamos kurti sistemos prototipas, sukurtas tikslams, aprašytiems skyriuje „Keliama problema ir projekto tikslas“.

Realizuotas prototipas buvo įdiegtas Raseinių miesto centrinėje katilinėje, atskiroje mašinoje (tam, kad netrikdyti veikiančios sistemos darbo). Prototipe realizuota realios vizualizacijos sistemos dalis.

Prototipo funkcionalumo ribos

Tam, kad būtų galima atlikti eksperimentą, buvo apibrėžtos ir realizuotos prototipo funkcionalumo ribos. Kadangi realaus objekto vizualizacijos apimtis yra labai didelė nebuvo tikslinga kurti visišką analogą. Tam kad būtų galima ištestuoti sistemos funkcionalumo privalumus ir trūkumus, buvo realizuotos vizualizacijos dalys, pateiktos *lentelė 7* lentelėje.

lentelė 7. Realizuotas vizualizacijos sistemos funkcionalumas

Sistemos dalis	Kas realizuota	Funkcionalumas
Įvykiai	Įvykių išklotinė	Leidžiama patvirtinti įvykį. Jei įvykio atsiradimo priežastys pašalintos, patvirtinus įvykį jis panaikinamas iš sąrašo. Jei priežastys, sukėlusios įvykio pasirodymą sistemoje vis dar aktyvios, patvirtintas įvykis specialiai pažymimas, o išnykus atsiradimo priežastims pašalinamas iš įvykių sąrašo
	Įvykio atvaizdavimas scenoje, pakeičiant objektų spalvą iš natūralios į raudoną (spalvų pakaitinis mainymas dėmesio atkreipimui)	Įvykus įvykiui, atitinkamas objektas scenoje pakeičia spalvą tam, kad būtų gaima kuo greičiau lokalizuoti ir suvokti įvykio kilmę, svarbą ir galimas pasekmes. Patvirtinus įvykį arba išnykus įvykį sukėlusioms priežastims, objektui scenoje grąžinama senoji spalva/tekstūra
3MW galios „verdančio sluoksnio“ vandens šildymo katilas	Vandens šildymo katilo trimatis modelis, atitinkantis realiai veikiantį įrenginį. Visi duomenys atvaizduojami tiksliai tose vietose, kur yra sumontuoti davikliai	--
Vandens šildymo katilą aptarnaujantys periferiniai įrenginiai	Dūmsiurbė	Tikslus modelis su duomenimis
	Orapūtė	Tikslus modelis su duomenimis
	Kuro padavimo šnekai	Tikslus modelis su duomenimis
	Termofikacinio vandens siurbliai	Tikslus modelis su duomenimis

Tyrimas buvo atliktas realizuoto funkcionalumo ribose. Sudarant įvykių sąrašą buvo parinkti tokie įvykiai, kurie (patikrinus sistemos *log*) sistemoje įvykdavo rečiausiai. Tai padaryta tam, kad

būtų galima ištirti sistemos naudingumą lokalizuojant ir vertinant įvykį bei nustatant jo atsiradimo priežastis.

Eksperimento tikslas

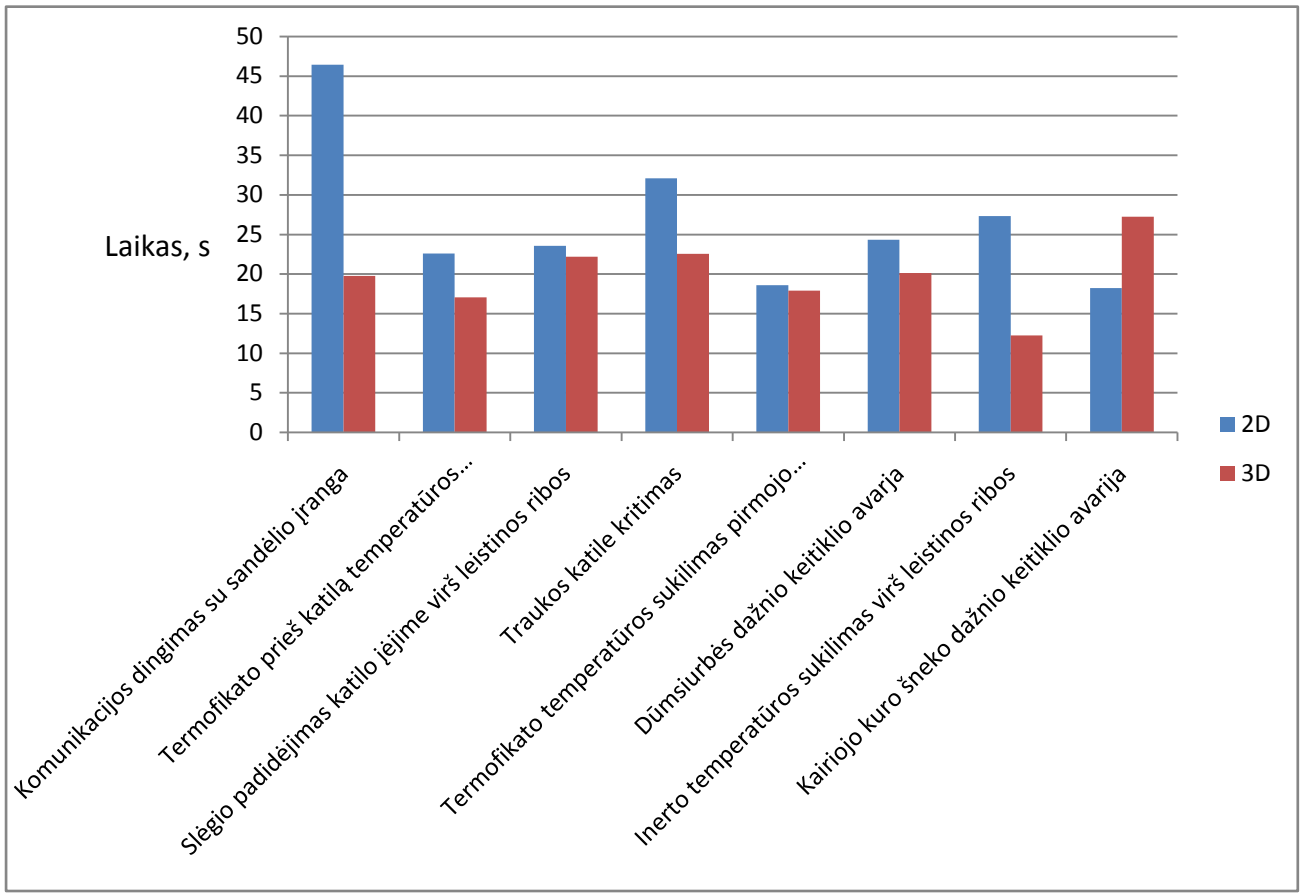
Eksperimonto tikslas buvo palyginti sukurtos sistemos prototipą su veikiančia dvimate vizualizacijos sistema. Buvo fiksuojamas vidutinis operatorių reakcijos laikas lokalizuojant sistemos įvykį ir jo atsiradimo vietą. Sukelti įvykiai:

- Komunikacijos dingimas su sandėlio įranga
- Termofikato prieš katilą temperatūros padidėjimas virš leistinos ribos
- Slėgio padidėjimas katilo įėjime virš leistinos ribos
- Traukos katile kritimas
- Termofikato temperatūros sukilimas pirmojo šilumokaičio išėjime
- Dūmsiurbės dažnio keitiklio avarja
- Inerto sluoksnio temperatūros sukilimas virš leistinos ribos
- Kairiojo kuro šneko dažnio keitiklio avarija

Kiekvienas iš išvardintų įvykių buvo sukeliamas tris kartus, kad būtų galima paimti reakcijos laiko vidurkį. Daugiau kartų sukelti tą patį įvykį yra neprasminga, nes tai per daug smarkiai iškraipytų rezultatus – operatorius pradeda atsiminti įvykį ir jo lokalizaciją bei priežastis.

Eksperimento rezultatai

Kadangi kiekvieno iš operatorių patirtis yra skirtinga, be to, skiriasi jų atsakomybės ribos, todėl ir rezultatams turėjo įtakos tai, kiek kuris operatorius (paprastai vyresnysis operatorius) praleisdavo laiko prie vizualizacijos sistemos. Toliau pateikiami rezultatai, atvaizduoti grafiku.


Laikas, s
50
45
40
35
30
25
20
15
10
5
0
2D
3D
Komunikacijos dingimas su sandėlio įranga
Termofikato prieš katilą temperatūros...
Slėgio padidėjimas katilo įėjime virš leistinos ribos
Traukos katile kritimas
Termofikato temperatūros sukilimas pirmojo...
Dūmsiurbės dažnio keitiklio avarija
Inerto temperatūros sukilimas virš leistinos ribos
Kairiojo kuro šneko dažnio keitiklio avarija

Pav. 19 test

Išvados

- Vizualizacijos sistemos kūrimas naudojant trimatę grafiką, yra sudėtingas ir laikui imlus procesas, reikalaujantis specifinių naudotojo žinių. Dvimatei vizualizacijai kurti yra sukurta daugybė standartinių vizualizacijos sistemų kūrimo paketų, ar programų šeimų. Tuo tarpu trimatės vizualizacijos kūrimui yra standartinės programavimo priemonės, kurių pagalba kai kuriuos uždavinius spręsti yra pakankamai sudėtinga.
- Trimatės sistemos derinimas yra sudėtingestis, nei dvimatės sistemos. Tai ypač pasireiškia tuomet, kada sistema dėl vienokių ar kitokių priežasčių „pakimba“ ir nebegalima iš vis dar veikiančio vizualizacijos proceso išgauti jokios informacijos, kuri galėtų paėti identifikuoti problemą.
- Tokių interaktyvios vizualizacijos sistemų pavyzdžių (naudojančių trimačius elementus) sukurta labai nedaug, lyginant su tradicine vizualizacijos sistema. Be to, didžioji dauguma iš jų sukurtos siauram ir labai konkrečiam tikslui atvaizduoti.
- Trimačius elementus naudojanti visa programinė įranga yra daug reiklesnė aparatūriniams resursams. Tai reiškia, kad sistemai efektyviai funkcionuoti reikia galingesnės, tuo pačiu ir brangesnės aparatūrinės įrangos.
- Rimtiems vizualizacijos projektams dažnai prireikia dviejų ar netgi naugiau monitorių vienu metu prijungtų prie tos pašios mašinos. Kaip minėta, trimatė grafika reikalauja gero aparatūrinio spartinimo. Šiuo metu laisvai įsigyjam grafinė įranga, palaikanti daugiau nei vieną monitorių, turi labai ribotas DirectX arba OpenGL technologijų aparatūrinio spartinimo galimybes, kas reiškia jog sistemos greitaveika ženkliai krenta, o sudėtingesnių trimačių scenų neleidžia realizuoti aparatūriniai apribojimai. Grafinė įranga, palaikanti kelis monitorius (*angl. multimonitoring*) ir turinti aparatūrinį DirectX arba OpenGL technologijų spartinimą yra nepaprastai brangi – to pasekoje paprasčiausiai neapsimoka kurti tokios vizualizacijos sistemos, kurios darbui reikalingos aparatūros kaina sudaro 20-30% vizualizacijos sistemos kainos.
- Trimatės grafikos elementus naudojanti vizualizacijos sistema yra estetiškai patrauklesnė už dvimatę. Kuriant dvimatę vizualizacijos sistemą, dažnei tenka transformuoti objektus į dvimatę erdvę iškreipiant jų tikrąjį vaizdą. Taip yra todėl, jog realybėje yra labai mažai objektų, kuriuos sudarantys elementai, pateikiantys vizualizacijos sistemai informaciją, būtų vienoje vizualizuojamo objekto pusėje. To pasekoje iškraipomas vizualizuojamo objekto vaizdas, kas nėra labai gerai – tai gali įnešti „sumaišties“ į vizualizacijos sistemą ir ją tampa sunkiau eksploatuoti. Naudojant trimačius elementus, visus vizualizuojamus objektus galima

atvaizduoti tiksliai tokius, kokie jie yra realybėje, be jokių iškraipymų. Praktiškai kiekvienas operatorius, išbandęs sistemą atkreipė į tai dėmesį ir įvertino kaip rimtą privalumą.

- Tikslus (netransformuotas ar kitaip neiškraipytas) vizualizuojamo objekto vaizdas sistemoje padeda daug geriau bei greičiau nustatyti sistemoje įvykusio įvykio kilmę ir jį lokalizuoti.
- Atlikus tyrimą galima teigti, jog sukurta sistema geriau perteikia informaciją apie sistemoje įvykusius įvykius ir grečiau padeda identifikuoti sutrikimo priežastis, tuo pačiu ir jas pašalinti.

Literatūra

[1] Vizualizacija [Žiūrėta 2009 04 13], prieiga internete http://lt.wikipedia.org/wiki/Vizualizacija

[2] Scientific visualization [Žiūrėta 2009 04 13], prieiga internete http://en.wikipedia.org/wiki/Scientific_visualization

[3] Knowledge vizualisation [Žiūrėta 2009 04 15], prieiga internete http://www.knowledge-visualization.org/

[4] Knowledge vizualisation [Žiūrėta 2009 04 15], prieiga internete http://www.knowledgevisualization.com/

[5] WIJK, J. Views on Visualization. *IEEE kompiuterinė duomenų bazė.* 2006 [žiūrėta 2008 11 04]. Prieiga internete: <http://ieeexplore.ieee.org/Xplore/guesthome.jsp>

[6] JEM, M. "THIN" vs. "FAT" VISUALIZATION CLIENT. *IEEE kompiuterinė duomenų bazė.* 1998 [žiūrėta 2008 11 10]. Prieiga internete <http://ieeexplore.ieee.org/Xplore/guesthome.jsp>

[7] ZHENG, L.; NAKAGAWA, H. *OPC (OLE for process control) specification and its developments.* Fukui, 2002. 917 – 920 p.

[8] The Industry's Foundation for High Performance Graphics [Žiūrėta 2007 11 07], prieiga internete http://www.opengl.org/

[9] JERN, M. 3D Data Visualization on the Web. *IEEE kompiuterinė duomenų bazė.* 1998 [žiūrėta 2007 11 08]. Prieiga internete: <http://ieeexplore.ieee.org/Xplore/guesthome.jsp>

[10] BRENT, D. et al. Designing a Visualization Framework for Multidimensional Data. *IEEE kompiuterinė duomenų bazė.* 2005 [žiūrėta 2007 11 08]. Prieiga internete: http://ieeexplore.ieee.org

[11] WARE, C., Information Visualization: Perception for Design, 2004.

[12] ADAMS, P. Performance Comparisons of Visualization Architectures. *IEEE kompiuterinė duomenų bazė.* 2005 [žiūrėta 2007 11 10]. Prieiga internete: <http://ieeexplore.ieee.org>

[13] RHYNE, T., et al. *Can we determine the top unresolved problems of visualization?* North Carolina, 2004. 563-565 p.

[14] GRESH, D.; KELTON, E. Visualization, Optimization, and Business Strategy: A Case Study. *IEEE kompiuterinė duomenų bazė.* 2003 [žiūrėta 2007 11 08]. Prieiga internete: <http://ieeexplore.ieee.org/Xplore/guesthome.jsp>

[15] WONG, P. et al. Dynamic Visualization of Transient Data Streams. *IEEE kompiuterinė duomenų bazė.* 2003 [žiūrėta 2007 11 04]. Prieiga internete: <http://ieeexplore.ieee.org/Xplore/guesthome.jsp>

[16] Centre for intelligent machines [Žiūrėta 2008 11 23], prieiga internete http://www.cim.mcgill.ca/

[17] Information vizualisation [Žiūrėta 2009 04 24], prieiga internete http://en.wikipedia.org/wiki/Information_visualization/

[18] Bring OPC Client capabilities to your *VB.NET*, *C#*, *Visual Basic*, *C++* and *Delphi* applications quickly using a proven tool that has been available since 2000 and OPC Interoperability tested many times [Žiūrėta 2009 05 03], prieiga internete http://www.opcactivex.com/

[19] SHENG LIAN, H.; WEI ZHONG, Y.; KE QIN, H The insufficiency and improving methods of GA. *IEEE kompiuterinė duomenų bazė.* 2002 [žiūrėta 2008 11 08]. Prieiga internete: http://ieeexplore.ieee.org

Santrumpų ir terminų žodynas

SCADA – Supervisory Control And Data Acquisition. Vizualizacijos sistemos.

Trimatis elementas – grafinis komponentas, atitinkantis vizualizuojamą objektą ir galintis keisti poziciją bei pasisukimo kampą erdvėje.

PLV – programuojamas loginis valdiklis. Tai įrenginys, paprastai skirtas fiziniams procesams valdyti. Gali būti laisvai programuojamas, turėti komunikacinius protokolus, leidžiančius duomenų mainus su išore.

WPF – Windows Presentation Foundation. Korporacijos Microsoft technologija, skirta vartotojo sąsajos kūrimui.

OPC – OLE for Process Control. Tai duomenų mainus užtikrinantis serveris, skirtas duomenų surinkimui/rašymui į procesą valdančius įrenginius. Realizuoja daug skirtingų komunikacinių protokolų ryšiui su įvairių gamintojų įranga užtikrinti. Duomenis atiduoda per standartizuotą sąsają.

Tag – duomenų žymė. Duomenų žymės kuriamos OPC serveryje tam, kad būtų galima jų pagalba identifikuoti skaitomus/rašomus duomenis.