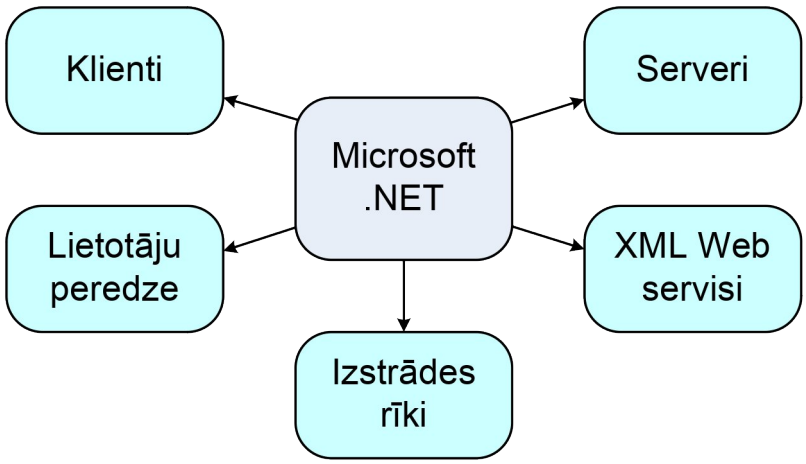
1. Ievads MS .NET Framework un MS Visual Studio

Šajā nodaļā tiks apskatītas .NET tehnoloģijas būtība, īpatnības un darbības principi, programmas produktu izstrādes pieejas, kā arī Microsoft (turpmāk tekstā MS) Visual Studio izstrādes vides īss raksturojums un darba sākums ar šo vidi.

1.1. MS .NET un .NET Framework apskats

MS .NET ir stratēģija, kas paredzēta labākai sistēmu mijiedarbībai organizācijas ietvaros. Tā sastāv no 5 pamata komponentiem (1.1.attēls).



1.1.attēls. MS .NET sastāvdaļas

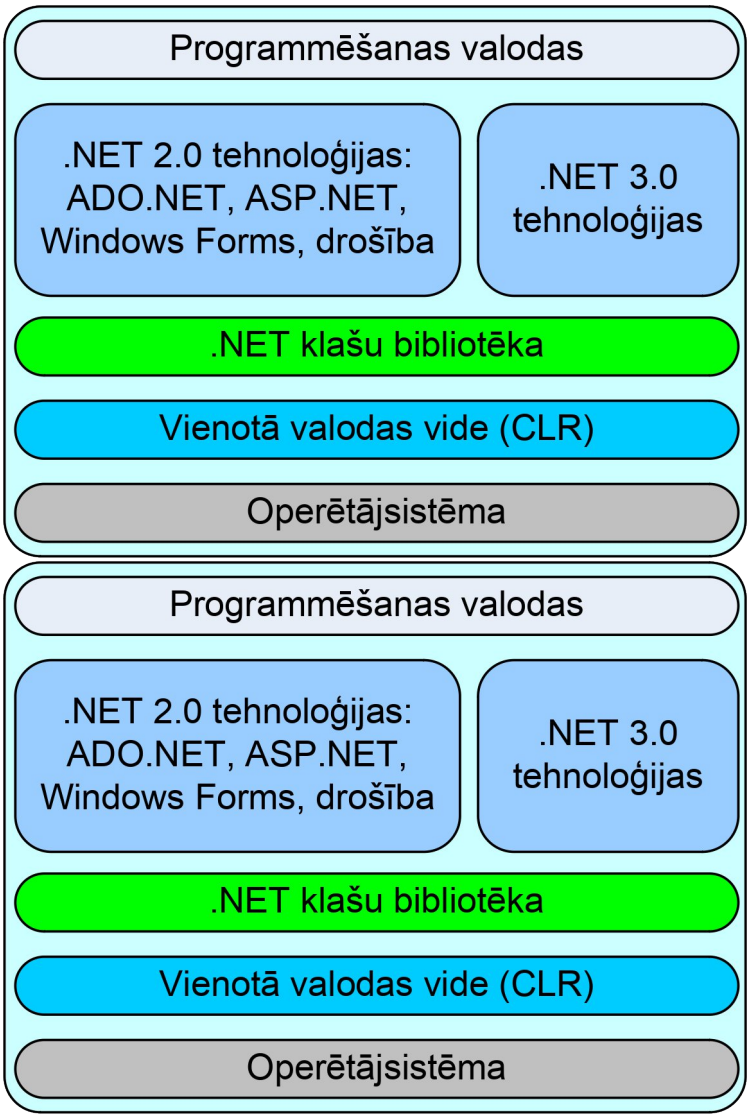
Apskatīsim katru no šīm sastāvdaļām:

- **Izstrādes rīki** – paredzēts aplikāciju izstrādei .NET vidē (viena no tiem ir MS Visual Studio, kura ļauj veidot dažādas aplikācijas personāliem datoriem un mobilām ierīcēm).
- **Klienti** izmanto izstrādātās aplikācijas; tie var būt dažāda veida (mobilās ierīces, Internet pārlūkprogrammas vai personālie datori);
- **Serveri** - .NET tehnoloģija piedāvā infrastruktūru aplikāciju veidošanai serveriem;
- **XML web servisi** (tīmekļa pakalpes) – iespēja izstrādāt savus web servisus;
- **Lietotāju pieredze** - .NET dod iespēju izmantot vienādus rīkus parastu un Internet aplikāciju izstrādei, kas ir daudz ērtāk no lietotāju puses.

Tīmekļa pakalpes (web services) - vienots veids, kā tīkla lietojumprogrammām, arī tādām, kas strādā dažādās platformās, rakstītas dažādās valodās u.t.t., savstarpēji sazināties.

Ziņu pārraidei tiek izmantoti starptīkla protokoli. Tīmekļa pakalpes nodrošina informācijas apmaiņas iespēju, nepārzinot otras puses sistēmas un programmatūru. Pakalpes lietotājam uzņēmumam ir svarīgi tikai saņemt konkrētu funkcionālu resursu, savukārt pakalpes sniedzējs var jebkurā brīdī brīvi izvēlēties, kādā valodā un platformā izstrādāt un darbināt nepieciešamos algoritmus.

.NET Framework ir tehnoloģija, kura ļauj izstrādāt aplikācijas uz .NET stratēģijas pamata. Tā sastāv no divām pamata komponentēm, kuras vienkāršo sadalīto aplikāciju izstrādi – **vienotā valodas vide** (*Common Language Runtime – CLR*) un **.NET Framework klašu bibliotēka** (skat. 1.2.attēlu).



1.2.attēls. .NET Framework struktūra

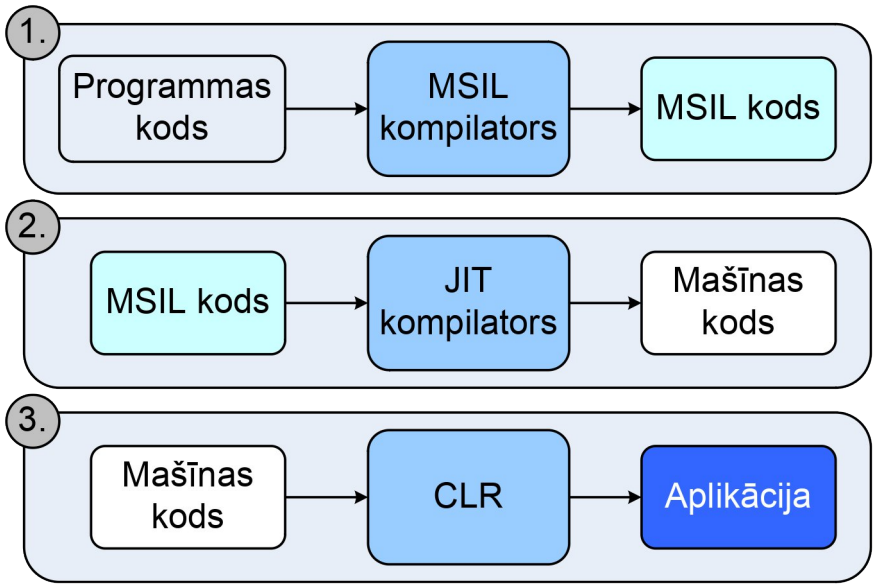
.NET tehnoloģija paredz vairāku programmēšanas valodu izmantošanu vienā un tajā pati izstrādes vidē:

- **Visual C#** (jauna jaudīgā objektorientētā valoda, kas izveidota uz C, C++ un Java valodu bāzes);
- **Visual Basic** (pēdēja Visual Basis valodas versija);
- **Visual C++**;
- **Visual J#** (valodas Java Microsoft versija, kura ļauj Java programmētājiem strādāt .NET vidē);
- **citas valodas** (nav iekļautas .NET Framework sastāvā, bet tās izmantošana ir pieļaujama).

.NET 2.0 un 3.0 tehnoloģijas sīkāk tiks apskatītas tālāk. Šīs tehnoloģijas ir pilnībā savietojamas.

.NET Framework klašu bibliotēka satur lielu klašu kolekciju, kuru var izmantot .NET aplikāciju izstrādē. Klašu funkcionalitāte aptver dažādas sfēras, piemēram, ADO.NET – ārējo datu avotu izmantošanai, ASP.NET – web aplikāciju izstrādei, Windows Forms – Windows aplikāciju izstrādei.

Vienotā izstrādes vide ir .NET fundamentālā koncepcija, kura nodrošina dažādu programmēšanas valodu lietojumu vienā vidē. .NET kompilators pārveido augstā līmeņa programmas kodu vadāmā modulī, kuru CLR var izpildīt. .NET Framework iekļauj arī speciālo servisu resursu vadībai, kuru sauc **Garbage Collector** (atkritumu kontrolieris). Šīs serviss automātiski atbrīvo atmiņu no nevajadzīgiem objektiem, kas agrāk bija jādara manuāli.



1.3.attēls. .NET Framework darbības princips

1.3.attēlā parādīta .NET Framework tehnoloģijas darbības shēma. Aplikācijas izpildes procesu šajā gadījumā var sadalīt trijos posmos:

1. Programmas izejas koda kompilācija videi saprotamā valodā. Piemēram, .NET kompilators pārveido C# kodu speciālā starpvalodā (*Microsoft Intermediate Language – MSIL*), kura ir „dzimtā" .NET Framework videi.
2. Pārveidot MSIL kodu mašīnkodā, kuru saprot klienta dators. Šo procesu .NET Framework veic ar Just-In-Time (JIT) kompilāciju.
3. Tālāk mašīnas kods tiek izpildīts vienotā valodas vidē, aplikācijas kods ir ielādēts atmiņā un programma gatava darbam.

.NET Framework iekļauj sevī arī vairākas īpatnības, kuras ļauj paaugstināt aplikācijas drošību:

Tiesības – aplikāciju var lietot daudz lietotāju, kurus fiziski kontrolēt nav iespējams, tādēļ ir nepieciešams definēt dažādus pieejas līmeņus, kurus .NET vidē var definēt kā lietotāju grupas ar noteiktām tiesībām.

Tipu drošība – vienotā valodas vide nodrošina vienotu tipu sistēmu, kad aplikācija var piekļūt objektiem tikai viena noteiktā veidā, kas savukārt ļauj novērst vispārīgas programmēšanas kļūdas.

Drošības politika – likumu kopa, kuru izmanto vienotā valodas vide tiesību piešķiršanā aplikācijas kodam. Tas tiek panākts sākuma noskaidrojot koda turpmākas darbības un tikai pēc tam atkarībā no tiesībām atļaujot vai aizliedzot tās.

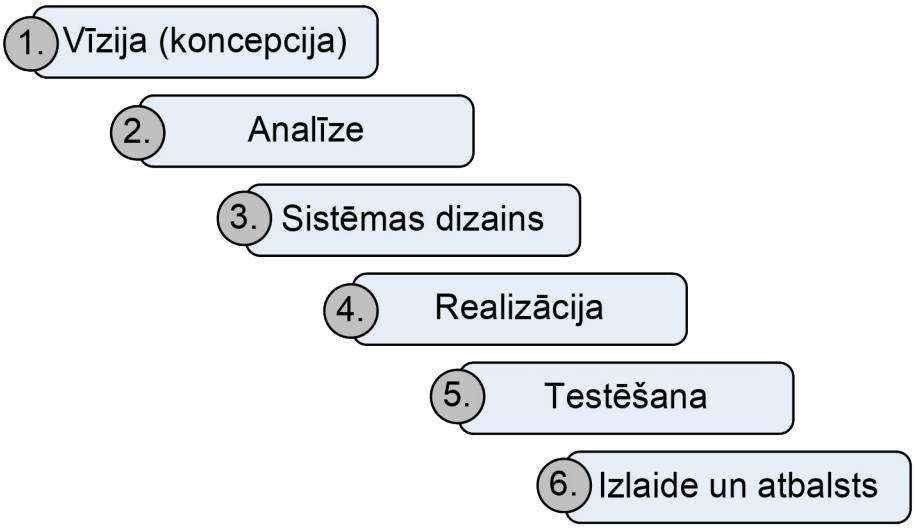
Koda pieejas drošība – atkarībā no koda avota (E-pasta piesaistne vai nezināmā tīmekļa lapa) tam var būt dažādas pieejas tiesības.

Lomu balstītā drošība – tikai lietotājs ar attiecīgu lomu var saņemt atļauju aizsargātām darbībām ar aplikāciju.

Win32 drošība tiek panākta ar operētājsistēmas drošību, ja operētājsistēma neļauj lietotājam veikt noteiktas darbības, tad darbs ar .NET aplikāciju arī nav iespējams.

1.2. Programmnodrošinājuma izstrādes cikls

Programmnodrošinājuma izstrāde pašlaik ir ļoti sarežģīts process, kurš iekļauj sevī vairākus posmus un prasa daudzu speciālistu iesaistīšanu. Shematiski to var attēlot šādi:



1.4.attēls. Programmnodrošinājuma izstrādes process

Vīzija – šajā posmā tiek definēti globālie projekta mērķi un uzdevumi.

Analīzes fāzē paplašina un konkretizē vispārīgus mērķus un uzdevumus no vīzijas fāzes, nosakot projekta (programmas produkta) funkcionālās prasības un ierobežojumus. Rezultātā tiek sagatavota **prasību specifikācija** ar detalizēto prasību aprakstu un **testa specifikācija** ar testiem (paraugiem) programmas pārbaudei.

Sistēmas dizaina fāze dokumentē projekta tehnisko specifikāciju, kur detalizēti apraksta projekta mērķu un uzdevumu sasniegšanai nepieciešamo sistēmas arhitektūru (piemēram, lietotāja saskarnes uzmetumi un biznesa loģikas apraksts pseidokodā).

Realizācijas (implementācijas) fāzē izveido gatavu programmas produktu balstoties uz tehnisko specifikāciju. Rezultātā jābūt gatavam programmas izejas kodam.

Testēšana paredz programmas pārbaudi ar testa specifikācijā noteiktiem paraugiem, atrasto neapstrādāto izņēmumu situāciju konstatēšana un labošana.

Izlaišanas un atbalsta fāzē programmnodrošinājumu sagatavo un nosūta klientam. Izplatīšanai gatavs programmnodrošinājums var saturēt:

- izpildāmā aplikācija, kuru klients var instalēt;
- lietotāja dokumentācija;
- sākotnējā lietotāju apmācība.

Pastāv dažādas pieejas programmnodrošinājuma izstrādei vai **izstrādes dzīves cikla modeļi**: ūdenskrituma, spirāles, pieauguma un V-formas modeļi.

Tabula 1.1.

Programmnodrošinājuma izstrādes modeļi

Modelis	Apraksts	Piemērotība	Priekšrocības	Trūkumi

Ūdenskrituma	1.4.attēla 6 soļu secīga izpilde, kad nākoša fāze sākas tikai pēc iepriekšējās fāzes pabeigšanas un apstiprināšanas.	Mazi projekti, kur prasības ir skaidras un nemainīgas.	Viegli pārbaudīt katru posmu.	Neelastīga, dārga izmaiņu gadījumos.
Spirāles	Fāzes ir reprezentētas spirāles veidā. Katrs spirāles aplis attēlo kādu izstrādes procesa posmu: - mērķu nostādne, - riska novērtēšana un samazināšana, - izstrāde un pārbaude, - plānošana.	Lieli projekti ar augstu riska pakāpi.	Elastīga, jo var lietots kopā ar citiem modeļiem. Orientēta uz risku.	Augsta personāla kvalifikācija, dārga.
Pieauguma	Projekts tiek izstrādāts iterāciju veidā, kad katra iterācija iekļauj sevī analīzes, dizaina un realizācija fāzes.	Projekti ar bieži gaidāmām izmaiņām.	Maksimālā klientu iesaistīšana un to vēlmju apmierināšana.	Grūti pārbaudīt katru posmu, dārga
V-formas	Līdzīga ūdenskrituma modelim, katrai fāzei papildus pievienojot testa fāzi (piem., prasību specifikācijai ir nepieciešami apstiprinājums un sistēmas testēšana).	Projekti ar labi definētiem un skaidriem mērķiem.	Viegla un vienkārša pārbaudē	Neelastīga, dārga prasību izmaiņu gadījumos.

1.3. Ievads MS Visual Studio

MS Visual Studio ir integrētā izstrādes vide, kura ļauj veidot .NET aplikācijas dažādās programmēšanas valodās. Tā iekļauj sevī vairākas daļas:

Windows formu veidošanas rīki iekļauj sevī:

- sākotnējo formu šablonu, kur var izvietot aplikācijas kontroles (vadīklas);
- rīkjoslu ar iepriekš definētām kontrolēm;
- grafisko vidi, kur var veidot Windows formas;
- testa vidi aplikācijas izpildei un atkļūdošanai.

Līdzīgus rīkus piedāvā arī **Web formu veidošanas rīki** un **XML tīmekļa pakalpes veidošanas rīki**.

.NET aplikāciju veidošanas rīki ir:

- Windows Communication Foundation – sadalīto aplikāciju uzlabotais izstrādes rīks;
- Windows Presentation Foundation – paredzēts lietotāja saskarnes izstrādei (iespēja ar XML palīdzību aprakstīt saskarni utt.);
- Windows Workflow Foundation – izstrādes un citu procesu attēlojums;
- Windows Cardspace – iespēja pielietot drošībai virtuālās biznesa vietkartes bez lietotāja vārda un paroles lietošanas nepieciešamības.

Datu pieejas komponenti nodrošina pieeju datu bāzēm no .NET aplikācijas. Visplašāk tiek izmantoti ADO.NET komponenti dažāda tipa datu piekļuvei.

MS Visual Studio izmanto **risinājumus** (solutions) un **projektus** (projects) kā konceptuālus konteinerus izejas koda organizēšanai programmnodrošinājuma izstrādes laikā, kas vienkāršo sarežģītu projektu izstrādi.

Risinājumā var būt viens vai vairāki projekti, kā arī neatkarīgās vienības, kurus projekti izmanto. Risinājuma datnēm ir divu veidu paplašinājumi:

.sln - standarta risinājuma datne, kura ļauj atvērt risinājumu ar visiem tās sastāvdaļām;

.suo - Visual Studio iestatījumi, kuri attiecas uz konkrēto risinājumu.

Projekti paredzēti izejas koda datņu, referenču un projekta līmeņa konfigurācijas iestatījumu organizēšanai. Projekta izveidošanas laikā tās automātiski tiek organizēts risinājumā. Plašāk lietojamās projekta datnes ir (programmēšanas valodas Visual C# gadījumā):

.cs – izejas koda datne, kas pieder projektam un var reprezentēt moduli, Windows formas datni vai klases datni;

.csproj – konkrētā projekta datnes;

.aspx – ASP.NET tīmekļa lappuses datnes;

.asmx – ASP.NET tīmekļa pakalpes (web services).

Nodaļas kopsavilkums

MS .NET ir stratēģija, kas paredzēta labākai sistēmu mijiedarbībai organizācijas ietvaros. Tā sastāv no 5 pamata komponentiem: serveri, klienti, lietotāju pieredze, izstrādes rīki un XML tīmekļa pakalpes.

.NET Framework ir tehnoloģija, kura ļauj izstrādāt aplikācijas uz .NET stratēģijas pamata. Tā sastāv no divām pamata komponentēm, kuras vienkāršo sadalīto aplikāciju izstrādi – vienotā valodas vide un .NET Framework klašu bibliotēka. .NET sastāvā ir tehnoloģijas datu pieejai (ADO.NET), tīmekļa lappušu izstrādei (ASP.NET), Windows formu izstrādei un drošības līdzekļi.

.NET Framework ļauj programmēt vairākās programmēšanas valodas (Visual C#, Visual C++, Visual J#, Visual Basic un citi) vienotā vidē.

Programmnodrošinājuma izstrāde iekļauj sevī sešas fāzes: vīzijas, analīzes, sistēmas dizaina, realizācijas, testēšana un izlaišanas un atbalsta fāzes. Izstrādes process var būt realizēts izmantojot dažādus izstrādes modeļus: ūdenskrituma, spirāles, pieauguma un V-formas.

MS Visual Studio ir integrētā izstrādes vide, kura ļauj veidot .NET aplikācijas dažādās programmēšanas valodās. Tā iekļauj sevī: Windows formu veidošanas rīkus, Web formu veidošanas rīkus, XML tīmekļa pakalpes un .NET veidošanas rīkus, vairāku .NET programmēšanas valodu atbalsts, datu pieeju, izņēmuma situāciju apstrādi, palīdzību un dokumentāciju.

UZDEVUMI PAŠPĀRBAUDEI

1. Kādas ir .NET Framework sastāvdaļas un kādas īpatnības tām piemīt?
2. Raksturojiet .NET aplikācijas kompilācijas procesu.
3. Aprakstiet programmnodrošinājuma izstrādes modeļus.
4. Kādas MS Visual Studio īpatnības Jūs zināt?