Дәріс 14

UML тілі және объектіге бағытталған талдау, диаграммалардың әртүрлілігі. Кластардың диаграммалары.

Лекция жоспары:

1. Объектілік ұстанымға негізделген программалық жабдықтардың ерекшеліктері және оларға қойылатын талаптар
2. Модельдеу тілі
3. UML –де қолданылатын негізгі диаграммалар
4. Класстар диаграммасы

Лекция мазмұны

1. Объектілік ұстанымға негізделген программалық жабдықтардың ерекшеліктері және оларға қойылатын талаптар

Объектіге бағдарланған программалау – бұл программалық жабдықты, қандай да болмасын кластың өкілі болып табылатын, объектілердің жиынтығы түрінде құратын программалау методологиясы.

Объектіге бағдарланған жобалау – бұл құрылатын ақпараттық жүйенің (немесе программалық жабдықтың) барлық статикалық және динамикалық модельдерін объектілі декомпозициялау процесі мен модельдердің логикалық, физикалық тұрғыдан беру тәсілдері негізінде жобалау методологиясы.

Объектіге бағдарланған талдау – бұл жобаланатын жүйеге қойылатын талаптар, пәдік облыстағы анықталған кластар мен объектілер тұрғысынан қарастырылатын методология.

Объектіге бағдарланған ұстанымның концептуалдық негіздеріне объектіге бағдарланған ұстанымның моделі жатады. Объектілік модельдеудің негізгі элементтері: абстаркциялау, инкапсуляция, модульділік және иерархия. Қосымша элементтері: типтелу, паралеллизм және тұрақтылық. *Абстракциялау* – бұл қандай да болмасын объектіні, өзге объектілерден ажырататын белгілері, сипаттамалары және т.б. арқылы бөліп алу, жалпы абстракциялау объектінің сыртқы ерекшеліктеріне негізделеді. Объектіге бағдарланған ұстанымда, берілген объектінің дұрыс абстракциялануы, жобалаудың негізгі міндеттерінің бірі болып саналады. *Инкапсуляция* – бұл объектінің, өзінің ішкі элементерінің, бір бірінен ажыратылу процесі. Бұл процесс кезінде объектінің ішкі құрылымдары мен оқиғалары бір- бірінен дұрыс ажыратылады. Инкапсуляция объектінің интерфейсін қорғау үшін қолданылады немесе объектілік ұстанымда класстың ресурстарын, тек оның өзінің ғана пайдалануын қолдайды.

Абстракциялау мен инкапсуляция бірін бірі толықтырады.

Модульділік – бұл программалық жабдықтың декомпозициялану кезінде өзара байланысқан, бірақ өте әлсіз байланысқан модульдерге бөліну қасиеттері. Инкапсуляция мен модульділік қасиеттері абстракцияларды бір- бірінен ажыратады.

Иерархия – бұл жүйедегі абстракцияланудың бір- біріне бағынышты түрде реттеліп орналасуын тағайындайды. Бұл күрделі жүйедегі класстардың құрылымы (иерархиясы). Мысалы, жай және көп қабылдаушылықты айтуға болады.

Типтелу – бұл абстракцияға байланысты класстарды бір бірінен ажырату үшін қойылатын шектеулер.

Параллелизм – бұл объектінің актив және пассив түрде болуын көрсетеді.

Тұрақтылық – бұл объектінің өмір сүру уақытын көрсетеді.

Объектіге бағдарланған ұстанымның негізгі түсініктері: объект, класс.

Объект класстың экземпляры тұрғысынан қарастырылады. Объектінің күйі, оқиғасы және жеке қасиеттері болады. Объектіге әсер етуді әдіс деп атайды. Класс қабылдаушылық пен инкапсуляция және полиморфизмді (абстракцияны) қанағаттандыратын құрылымдық жиынтық тип ретінде қабылданған..

Объектіге бағдарланған анализ бен жобалау әдістері модельдеу тілі мен модельдеу процестерінің сипаттамаларынан тұрады.

2. Модельдеу тілі

Модельдеу тілі жобаның сипаттамасын беру үшін қолданылатын нотация. Нотация – бұл модельдерде қолданылатын графиктік объектілердің жиынтығы. Модельдеу тілінің синтаксисі де нотациямен анықталады. Процесс – бұл жобаны құру кезінде жасалатын қадамдардың сипаттамалары.

UML (Unified Modeling Language) – бұл 1980-1990 ж. қолданылып келген, объектіге бағдарланған анализ бен жобалаудың орнына келген әдіс болып табылады. UML алу үшін бірнеше авторлардың әдістерін біріктіруге тура келді: Boosh – авторы Гради Буч; OMT (object modeling technique) – авторы Джеймс Рамбо; OOSE (object oriented SoftWare engineering) – авторы Ивар Якобсон.

UML тілінің негізгі мақсаттары мен мүмкіндіктері:

- қолданушыға түсінікті болатын визуальды модельді құру;
- модельдегі базалық концепциялардың кеңейтуге бейім болуы;
- программалау тілдеріне, құру процессіне тәуелсіз болуы;
- модельдеу тілінің формальды негізде болуын қамтамасыз етеді;
- объектілік бағдарланған жабдықтар нарығына стимуляция жасайды;
- практикалық тәжірибелердің ең жақсысын біріктіру және тарату; UML-дың пайда болу және даму тарихына сәйкес келесі нұсқалары белгілі (3.11- сурет. Википедия бойынша 2011 жылғы мәлімет):

Нұсқа	**Қабылданған уақыты**
1.1	ноябрь 1997[1]
1.3	март 2000[2]
1.4	сентябрь 2001[3]
1.4.2.	июль 2004[2]
1.5	март 2003[4]

2.0	июль 2005[5]
2.1	Заңды түрде қабылданбаған [2]
2.1.1	август 2007[6]
2.1.2	ноябрь 2007[7]
2.2	февраль 2009[8]
2.3	май 2010[9]
2.4 beta 2	март 2011[10]

Сурет 11 - UML-дың нұсқалары туралы деректер

3. UML –де қолданылатын негізгі диаграммалар

UML 1.4.2 нұсқасы халықаралық ISO/IEC 19501:2005 стандартының негізі болып саналады. UML –де қолданылатын негізгі диаграммаларды келесі топтарға бөліп қарастырады (3.12- сурет. Википедия бойынша 2011 жылғы мәлімет):

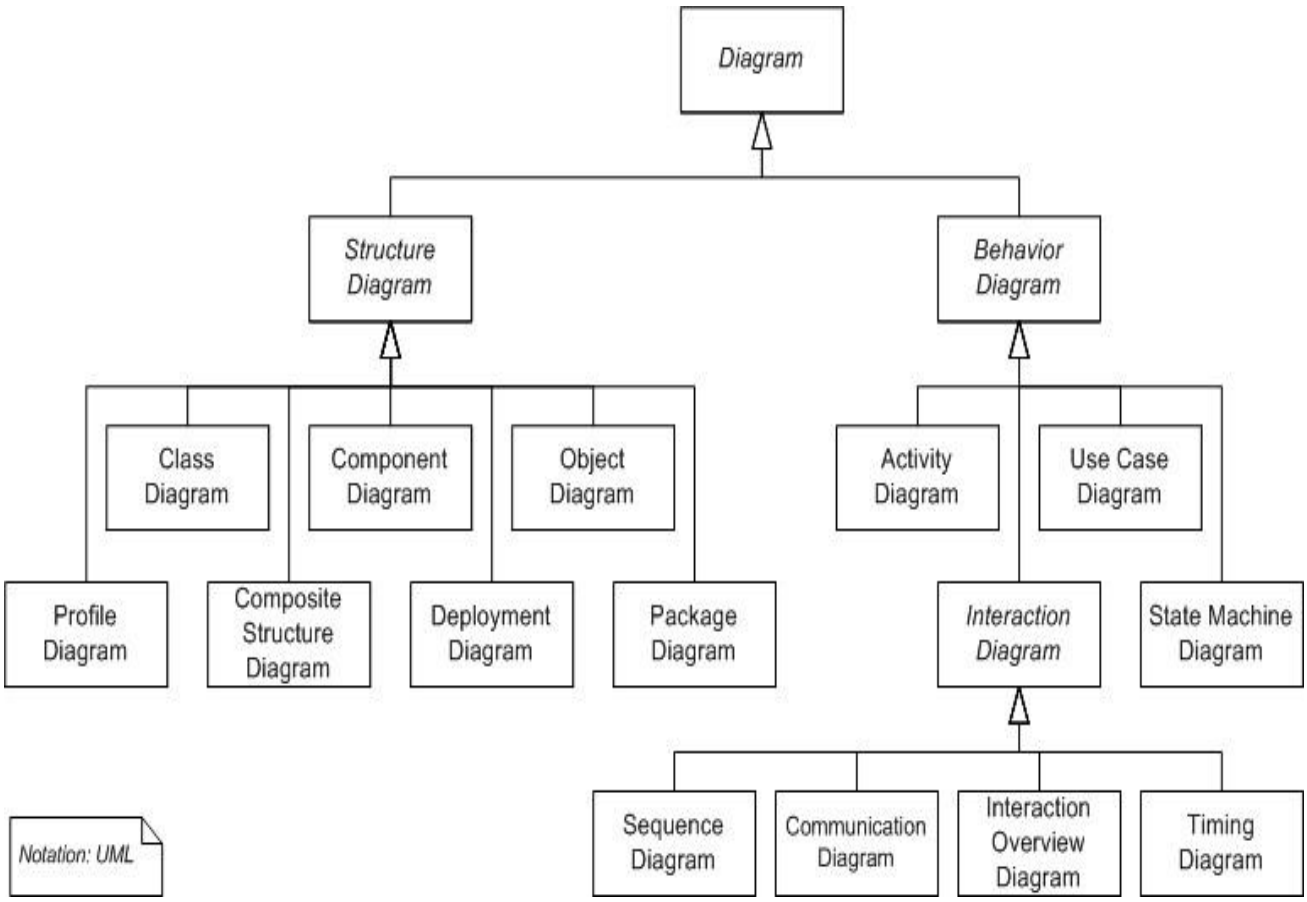
Құрылымдық диаграммалар	**Structure Diagrams:** *(Ағылшын тілінде)*	**Структурные диаграммы:** (*Орыс тілінде)*
Кластар диаграммасы	Class diagram	Диаграмма классов
Компоненттер диаграммасы	Component diagram	Диаграмма компонентов
Композит/ құрама құрылымдар: *Кооперация диаграммасы (UML2.0) немесе келісімді үйлестіруші диаграмма*	Composite structure diagram: *Collaboration (UML2.0)*	Композитной/составной структуры: *Диаграмма кооперации (UML2.0) или диаграмма сотрудничества*
Таратылған диаграмма	Deployment diagram	Диаграмма развѐртывания
Объектілер диаграммасы	Object diagram	Диаграмма объектов
Пакеттер диаграммасы	Package diagram	Диаграмма пакетов
Профильдер диаграммасы (UML2.2)	Profile diagram (UML2.2)	Диаграмма профилей (UML2.2)
Өзгеру	**Behavior Diagrams:**	**Диаграммы поведения:**
диаграммалары		
Қызмет диаграммасы	Activity diagram	Диаграмма деятельности

Күй ауысу диаграммасы	State Machine diagram	Диаграмма состояний
Прецеденттер диаграммасы	Use case diagram	Диаграмма прецедентов (вар-тов)
Өзара әрекеттесу диаграммасы: - *Коммуникация диаграммасы (UML2.0) / кооперация диаграммасы (UML1.x)* - *Өзара әрекеттесуді шолу диаграммасы (UML2.0)* - *Реттілік диаграммасы* - *Синхронизация диаграммасы (UML2.0)*	Interaction Diagrams: - *Communication diagram (UML2.0) / Collaboration (UML1.x)* - *Interaction overview diagram (UML2.0)* - *Sequence diagram* - *Timing diagram (UML2.0)*	Диаграммы взаимодействия: - *Диаграмма коммуникации (UML2.0) / Диаграмма кооперации (UML1.x)* - *Диаграмма обзора взаимодействия (UML2.0)* - *Диаграмма последовательности* - *Диаграмма синхронизации (UML2.0)*

Мысалы, UML 2.3 нұсқасында қолданылатын диаграммалар «құрылымын» UML- дегі «класстар диаграммасын» (class diagram) пайдаланып келесі түрде көрсетуге болады (3.13 - сурет).

Варианттар диаграммасы – бұл ұғымды Ивар Якобсон енгізген. Бұл варианттар диаграммасы жүйенің қандайда болмасын сыртқы әрекетіне жауап беруі. Әдетте, варианттар диаграммасы жүйе мен қолданушы арасындағы қатынасты сипаттайды. Программалық жабдықты жобалауда варианттар диаграммасы қолданушының қандай функцияларды орындау керектігін анықтау үшін жасалады. Мұнда байланыстың екі түрі қолданылады: USES (қолдану), EXTENDS (кеңейту)

CASE құралдар варианттар диаграммасында детализацияны әр түрлі деңгейде қолданады. Мысалы: Якобсон он адам бір жыл жасайтын жобалардағы варианттар саны 20-дан аспау керек деп есептейді.



Сурет 12 - Диаграммалар құрылымы

4. Класстар диаграммасы

Класстар диаграммасы – жүйедегі класстардың статикалық құрылымын модельдеу үшін және класстар арасындағы байланысты көрсету үшін жасалады.

Класстар диаграммасы объектіге бағдарланған ұстанымдағы негізгі диаграмма болып табылады. Класстар диаграммасының қызметі: жүйедегі объектілердің типін анықтау және олардың арасындағы байланысты көрсету болып келеді. Байланыстың статикалық екі түрі қолданылады: ассосация және подтиптер (тума типтер).

Бұлардан басқа класстар диаграммасының элементтеріне атрибуттар, операциялар және объектілер арасындағы шектеулер жатады. Класстар диаграммасын жобалаудан бұрын, ол диаграмманың қандай мақсатта қолданылатынын анықтап алу керек.

Класстар диаграммасын жобалаушы үш түрлі мақсатта қолдануы мүмкін:

- *концептуалдық аспект* – мұнда класстар даграммасы зерттелетін пәндік облыстағы негізі ұғымдарды анықтайды. Бұл ұғымдар болашақта құрылатын класстарға сәйкес болу керек, бірақ іс жүзінде ол барлық уақытта бірдей орындалмайды. Сондықтан концептуалдық модель болашақ ақпараттық жүйемен әлсіз байланыста болады және ол программалау тіліне тәуелсіз болады;

- *спецификациялық аспект* – мұнда құрылатын диаграмма ақпараттық жүйенің (программалық жабдықтың) интерфейсі деңгейінде жасалады. Класстың өзінің ішкі құрылымы қарасытырылмайды;
- *жүзеге асыру аспектісі (реализация)* – мұнда класстар диаграммасы ақпараттық жүйеге (программалық жабдыққа) қатысатын класстарды ішкі құрылымдарымен қоса анықтайды. Бұл аспекті программистер үшін негізгі диаграмма болып табылады.

Бақылау сұрақтары

1. UML тілі және объектіге бағытталған талдау?
2. Диаграммалардың әртүрлілігі?
3. Кластардың диаграммалары?
4. Объектілік ұстанымға негізделген программалық жабдықтардың ерекшеліктері және оларға қойылатын талаптар?
5. Модельдеу тілі?
6. UML –де қолданылатын негізгі диаграммалар? **7.** Класстар диаграммасы?

Ұсынылатын әдебиеттер

1. Роджерсон Д. Основы COM. Microsoft Press. 1997.
2. Скотт К. UML. Основные концепции. – М. 2002.
3. Чеппел. Технология ActivX и OLE. Microsoft Press. 1997.
4. Шилд Г. Самоучитель C++. BHV – Санкт Петербург. 1996.

Шилд Г. Теория и практика C++. BHV – Санкт Петербург. 1996