

Object-Oriented Analysis

Օբյեկտ կողմնակա և օբյեկտ-որիենտացիոն
հաշվազում

Գրեյի Բուչի մեթոդոլոգիան և օրինակներ

Քառսի առաջացումը

Բժիշկը, ինժեները և համակարգչային գիտնականը վիճում էին, թե որն է աշխարհի ամենահին մասնագիտությունը: Բժիշկն ասաց.

«Աստվածաշնչում գրված է, որ Աստված Եվային ստեղծեց Ադամի կողից վերցված մի կողոսկրից: Սա ակնհայտորեն վիրահատությունն էր պահանջում, ուստի ես կարող եմ ճիշտ ասել, որ իմ մասնագիտությունն է աշխարհի ամենահինը»:

Քաղաքացիական ինժեները ընդհատեց նրան և ասաց. «Բայց ճնշող գրքում ավելի վաղ ասվում է, որ Աստված քառսից ստեղծեց երկնքի և երկրի կարգը: Սա քաղաքացիական ինժեներիայի առաջին և անկասկած, ամենահիասքանչ կիրառությունն էր: Հետևաբար, հարգելի բժիշկ, դուք սխալվում եք. իմն է աշխարհի ամենահին մասնագիտությունը»:

Համակարգչային գիտնականը թեքվեց աթոռին, ժպտաց և վստահ ասաց.

«Հա, բայց ձեր կարծիքով ո՞վ ստեղծեց քառսը»:



Բարդ համակարգերի օրինակներ



1. Բույսի կառուցվածքը

Բույսը կազմված է արմատներից, ցողունից և տերևներից:
Տերևը՝ բջիջներից, բջիջը՝ օրգանոիդներից:

Սովորեցնում է. Հիերարխիա և Տրոհելիություն: Մենք կարող ենք ուսումնասիրել բջիջը որպես առանձին միավոր:



2. Ժամագործների պատմությունը

Մի ժամագործը հավաքում էր ժամացույցը միանգամից
(ձախողվում էր), մյուսը՝ փոքր կայուն ենթամասերից:

Սովորեցնում է. Կայունություն: Բարդ համակարգը հնարավոր է ստեղծել միայն կայուն միջանկյալ ձևերի միջոցով:

Ինչու՞ է ծրագրային ապահովումը բարդ

1. Խնդրի տիրույթի բարդություն (Problem Domain)

Օգտատերը և ծրագրավորողը խոսում են տարբեր լեզուներով:

Օրինակ՝ Բանկիրը գիտի բիզնեսը, բայց ոչ կողը. ծրագրավորողը հակառակը:

2. Գործընթացի կառավարում (Process Management)

Հազարավոր կոդի տողեր և հարյուրավոր մարդկանց աշխատանք:

Օրինակ՝ Կառավարել այնպես, որ բոլորի գրածը սինխրոն աշխատի:

3. Ճկունություն (Flexibility)

Ծրագիրը «փափուկ» է, ինչը բերում է անվերջ փոփոխությունների:

Օրինակ՝ Նոր պահանջները անընդհատ քայքայում են սկզբնական ճարտարապետությունը:

4. Դիսկրետ համակարգեր (Discrete Systems)

Փոքր սխալը կարող է բերել աղետալի հետևանքների:

Օրինակ՝ Մեկ սխալ ստորակետը կամ բիթը կարող է կանգնեցնել ամբողջ համակարգը:



Բարդ համակարգի 5 հատկանիշները



Հիերարխիկ

Կազմված է փոխկապակցված
ենթահամակարգերից:

Համակարգիչ -> Մայր սալիկ -> Պրոցեսոր ->
Տրանզիստոր:



Հարաբերական

«Պարզ մասնիկի» ընտրությունը
սուբյեկտիվ է:

String-ը պարզ է ծրագրավորողի համար, բայց
բարդ՝ OS գրողի համար:



Տրոհելի

Ներքին կապերը ավելի ամուր են, քան
արտաքինները:

Սա թույլ է տալիս առանձնացնել մասերը
(Encapsulation):



Օրինաչափություններով հարուստ

Կան կրկնվող նախշեր տարբեր
մակարդակներում:

Օրինակ՝ Design Patterns (Singleton, Factory):



Կայուն

Չարգանում է պարզ ու աշխատող
համակարգերից:

Բարդ համակարգը չի ստեղծվում գրոյից, այն
էվոլյուցիայի արդյունք է:

| Հիմնական սահմանումներ

OOP(Object oriented programming) - Օբյեկտ կողմնորոշված ծրագրավորում

Օբյեկտ կողմնորոշված ծրագրավորումը, ծրագրերի իրականացման մեթոդ է, որտեղ ծրագրերը կազմակերպվում են որպես համագործակցող օբյեկտների հավաքածուներ, որտեղ յուրաքանչյուր օբյեկտ ներկայացնում է որոշակի կլասի(յուրաքանչյուր օբյեկտ ունի տիպ) օրինակ և որոնց կլասներն պատկանում են կլասների հիերարխիայի, որոնք միավորված են ժառանգական կապերով:

Չատկանշական են հետևյալ կետերը

- Օբյեկտ կողմնորոշված ծրագրավորումը օգտագործում է օբյեկտներ՝ կառուցելու համար ֆունկցիոնալ տրամաբանական բլոկներ ալգորիթմների փոխարեն
- Յուրաքանչյուր օբյեկտ որևէ կլասի օրինակ է
- Կլասները միմյանց հետ կարող են կապված լինել ժառանգական կապերով

Հիմնական սահմանումներ

OOA(Object oriented analysis) - Օբյեկտ կողմնորոշված վերլուծություն

Օբյեկտ կողմնորոշված վերլուծությունը(անալիզը), վերլուծության մեթոդ է, որը ուսումնասիրում կլասների(դասերի) և օբյեկտների այն պահանջները որոնք առկա են խնդրի տեսանկյունից:

Օրինակ - Գրադարանի ավտոմատացման համակարգ

1. Օբյեկտների բացահայտում: Մենք չենք նայում կոդին, այլ նայում ենք իրականությանը:
Գտնում ենք՝ Գիրք, Ընթերցող, Գրադարանավար:
2. Պահանջների սահմանում: Որոշում ենք՝ ի՞նչ պետք է անի Գիրքը: Այն պետք է ունենա «կարգավիճակ» (վերցրած է, թե ազատ):

Հիմնական սահմանումներ

OOD(Object oriented design) - Օբյեկտ կողմնորոշված նախագծում

Օբյեկտ կողմնորոշված նախագծումը, նախագծման մեթոդ է, որը ներառում է օբյեկտ կողմնորոշված բաժանման մեթոդը(դեկոմպոզիցիայի մեթոդը) և նշագրության գործընթաց(նկարելու, պատկերելու գործընթաց), որը նախատեսված է համակարգի ինչպես տրամաբանական և ֆիզիկական, այնպես էլ ստատիկ և դինամիկ մոդելները պատկերելու համար:

Օբյեկտ կողմնորոշված նախագծումը՝

- հանգեցնում է օբյեկտ կողմնորոշված դեկոմպոզիցիայի
- օգտագործում է տարբեր նշումներ՝ համակարգի տրամաբանական(կլասների և օբյեկտների կառուցվածք) և ֆիզիկական(մոդուլների և պրոցեսների համակցություն) նախագծման տարբեր մոդելները արտահայտելու համար, ինչպես նաև համակարգի ստատիկ և դինամիկ կոդերը:

| Object oriented design

Տրամաբանական (Logical)

Սահմանում է համակարգի ներքին կառուցվածքը՝ կենտրոնանալով դասերի (classes) և դրանց միջև գոյություն ունեցող կապերի վրա:

- Ցույց է տալիս "Ինչպիսի՞ն" է համակարգը իմաստային մակարդակում:
- Օգտագործվում են Class և Object դիագրամներ:
- Օրինակ: Reader-ը հանդիսանում է Person-ի տեսակ (Inheritance):

Ֆիզիկական (Physical)

Սահմանում է համակարգի իրականացման (implementation) ճարտարապետությունը՝ կենտրոնանալով ծրագրային մոդուլների և սարքավորումների վրա:

- Ցույց է տալիս, թե "Որտե՞ղ" և "Ինչպե՞ս" է կոդը տեղակայված սկավառակի կամ ցանցի վրա:
- Օգտագործվում են Module և Process դիագրամներ:
- Օրինակ: Գրադարանը կամ SQL Server-ի տվյալների բազան:

Ստատիկ (Static)

Նկարագրում է համակարգի կառուցվածքային հատկությունները, որոնք ժամանակի ընթացքում չեն փոփոխվում:

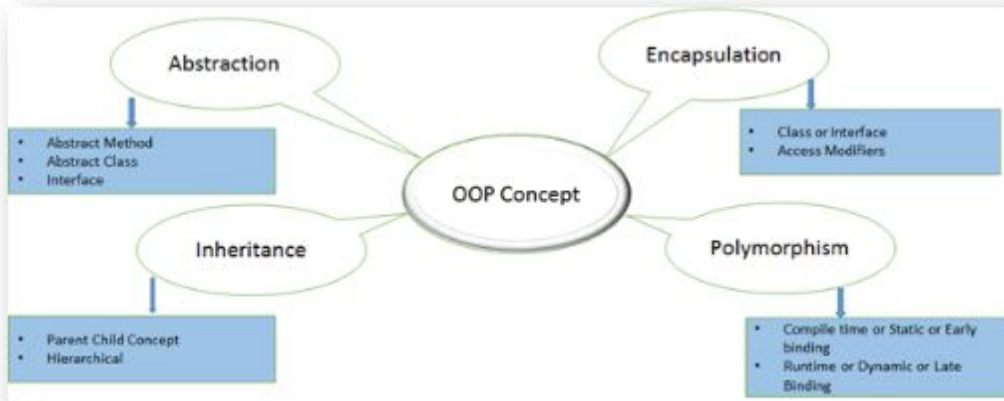
- Արտացոլում է համակարգի "Չանգիստ" վիճակը:
- Սահմանում է տվյալների բազան, դաշտերը և տիպերը:
- Օրինակ: Book դասը ունի ISBN (string) և Title (string) դաշտերը:

Դինամիկ (Dynamic)

Նկարագրում է համակարգի վարքագիծը (behavior) և օբյեկտների միջև հաղորդագրությունների փոխանակումը աշխատանքի ընթացքում:

- Ցույց է տալիս իրադարձությունների հաջորդականությունը և վիճակների փոփոխությունը:
- Օգտագործվում են Sequence և State Transition դիագրամներ:
- Օրինակ: Գրքի կարգավիճակի փոփոխությունը "Available"-ից "Borrowed":

Օբյեկտային մոդելի հիմնական տարրերը



Abstraction

Առանձնացնում ենք օբյեկտի միայն էական հատկանիշները:

Encapsulation

Թաքցնում ենք ներքին իրականացումը (implementation):

Modularity

Համակարգի տրոհումը փոխկապակցված միավորների:

Hierarchy

Աբստրակցիաների դասակարգում (Inheritance):