< Програм хангамжийн архитектур ба зохиомж >

(Лекцийн нэгтгэл)

Х. Очирсүх

ХШУИС, Програм хангамжийн 3 курсын оюутан, [20b1num0429@stud.num.edu.mn](mailto:20b1num0429@stud.num.edu.mn)

# Оршил

Энэхүү лекцүүдийн хүрээнд статик болон динамик ойлголтын харьцуулалт, холбоос, бүрдмэл болон нийлмэл холбоос, удамшил, багц, загварчлалын нэгдсэн хэлийг өргөтгөсөн механизмын талаар ойлголтыг авч, динамик болон статик ойлголтуудтай танилцаж, програм хангамжийн шинжилгээ болон зохиомжын, архитектур хэрхэн хийгдэх талаар судлаж пратикт хэрэгжүүлэх.

# Зорилго

Лекцүүдийн хүрээнд үзэх шаардлагатай агуулгуудыг судлаж, өөрийн үгээр тайлбарлаж тэмдэглэл бичих. Зорилгодоо хүрэхийн тулд дараах зорилтуудыг дэвшүүлсэн байгаа.

1. Лекцийн материал болон лекцийн үзэж ойлгох
2. Холбоос гэж юуг болохыг тайлбарлах
3. Холбоос класс болон бүлэглэсэн холбоос гэж юуг тайлбарлах
4. Бүрдмэл болон нийлмэл гэж ямар холбоос гэж юуг тайлбарлах
5. Удамшил гэж юу болох талаар тайлбарлах
6. Багц гэж юу болох талаар тайлбарлах

# Лекц 1-2

Програм хангамж гэж юу вэ?

Дан кодоос бүрддэггүй. Компьютерйин програм, код болон түүнд хамаарах бичиг баримтуудын цуглуулга юм.

## 

## Програм хангамжийн бүтээгдэхүүн гэж юу бэ?

Тодорхой шаардлагын дагуу бүтээгдсэн эцсийн програм хангамж юм.

Ерөнхийдөө

- Бүлэг хүмүүст худалдахаар бүтээгдэх

Хувиараа

- Нэг хэрэгчлэгчийн нарийвчласан шаардлагын дагуу

бүтээгдэх

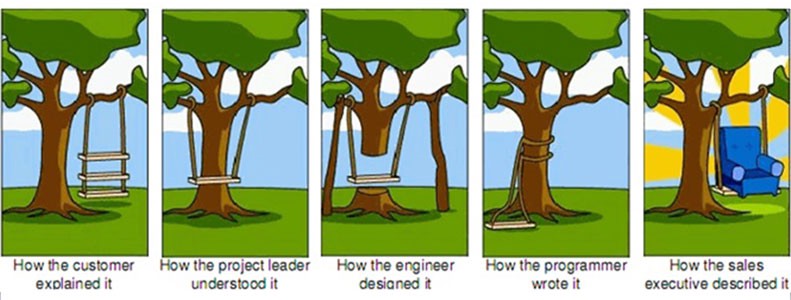
# Програм хангамжын хөгжүүлэлт яагаад хэрэгтэй бэ?

Програм хангамж бол нарийн эдрээтэй төвөгтэй байдлаараа онцлог байдаг.

Техник болон нийгэм ахуйн байдалд нөлөөлөх эдрээтэй байдал

1. Ямар ч нэг хүн програм хангамжын хамааралтай байдлыг цэгцэлж, өгөгдсөн хугацаанд амжиж хийх боломжгүй байдаг.
2. Програм хангамжын бүтээгдэхүүнийг өгөгдсөн хугацаанд хийх боломжгүй байдал ихээхэн үүсж байдаг.
3. Програм хангамжийн бүтээгдэхүүнийг бүтээхэд аливаа мэргэжилтний мэдлэг чадвар хэрэгтэй байдаг.
4. Програм хангамжын бүтээгдэхүүнийг бүтээхэд багаар ажиллах шаардлагатай байдаг.

Доорх зурган дээр харагдаж байгаагаар хэрэглэгчийн хүсэлт, шаардлагаас өөр бүтээгдэхүүнийг үүсгэсэн байгаа. Програм хангамжын хөгжүүлэлт ойлгомжгүй буруу тодорхойлж , ашиглахгүй байснаар хэрэглэгчийн хүссэн шаардлагад

нийцэхгүй бүтээгдэхүүний үүсгэх аюултай байдаг.

**Програм хангамжын хөгжүүлэлтэнд хөгжүүлэлтийн арга болон ойлголтууд шаардлагатай байдаг.**

# Програм хангамжын инженерчлэл гэж юу бэ?

Орчин үед дэлхийн дахинд компьютер шаардагдсан шинэ хэрэгцээ өдрөөс өдөрт ихсэж байна. Програм хангамж илүү нарийн ээдрээтэй болж байна. Жишээ хиймэл оюун ухаан бүтээхэд илүү нарийн програм бүтээх асуудлууд маш ихээр гардаг, хүн ба машины харилцаа илүү боловсронгуй болоход чиглэж байгаа учир програм хангамж маш нарийн боловсронгуй болж байна. Програм хангамжийн хэмжээ болон амьдрах хугацаа ихсэж байна. Өдрөөс өдөрт хэдэн мэдээллүүд хэдэн сая

килобайтаар хэмжигдэж байна. Програм хангамж систем хөгжүүлэлтийн салшгүй нэг хэсэг болж байна. Програм хангамжийг арчлаж тордох хөгжүүлэгчид ихсэж байна. Энэ бүгдээс харахад програм хангамжийн инженерчлэл маш чухал гэдгийг ойлгох хэрэгтэй.



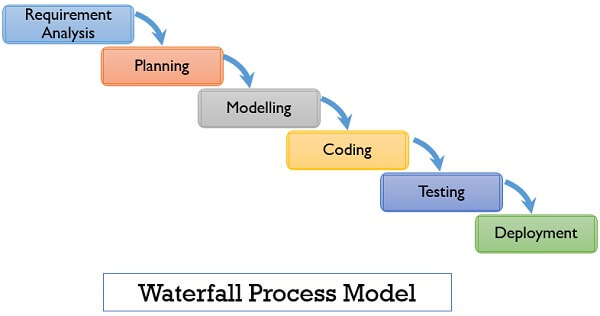
Хөгжүүлэлтийн процесс дараах зүйлсийг тодорхойлж өгдөг.

* Ажлын алхамууд
* Боловсруулагдсан баримт бичиг
* Програм хангамжын бүтээгдэхүүнийг боловсруулах дараалал
* Хөгжүүлэгчидийн багийн ажиллагаа

**Хүрхрээ загвар**

Тодорхой үйл ажиллагааг шугаман дараалсан үе шатад хуваадаг.

Програм хангамжаа боловсруулахад үзэл баримтлал эхлүүлэх, дүн шинжилгээ хийх, дизайн, туршилтийн үе шатууд нэг чиглэлд давхцахгүй тул энэхүү арга хялбар байдаг.

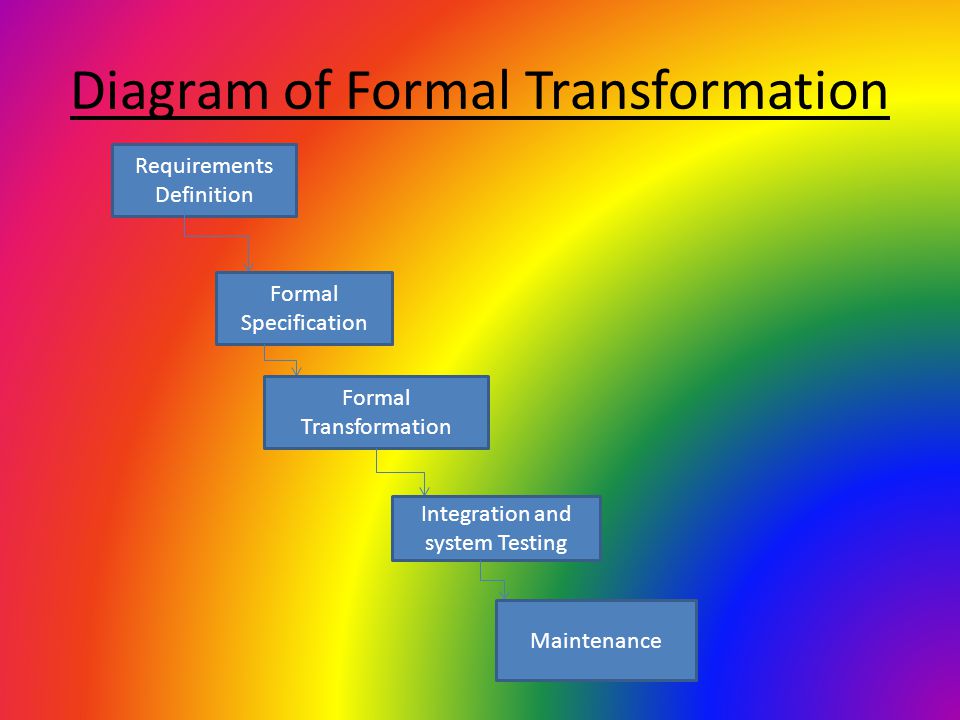


**Хувьсалттай хөгжүүлэх арга**

Програм хангамжийг хөгжүүлэх амьдралын мөчлөгийн давталт болон өсөлттэй загваруудыг хослуулсан. Системээ том тэсрэлтээр гаргах, цаг хугацааны явцад ахисан процессоор хүргэх нь энэ загварт хийгдсэн үйлдэл юм.

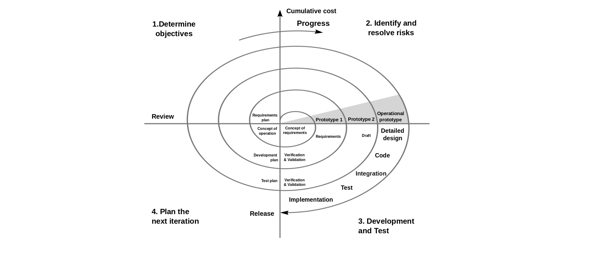


**Формал хувиргалт загвар**

Энэхүү загвар нь програм хангамжийн хөгжүүлэлтийг шаардлагуудыг албан ёсны (математик) тодорхойлолтоос эхэлж, аажмаар хэрэгжилт болгон хувиргах үе шатуудын дараалал юм.

**Spiral Model загвар**

Спираль загвар нь эрсдэлд тулгуурласан давталттай програм хангамжийн процессын загвар юм. Спираль загвар нь төслүүдийг гогцоонд хүргэдэг. Бусад процессын загваруудаас ялгаатай нь түүний алхамууд нь үйл ажиллагаа биш харин бүтэлгүйтлийн хамгийн их эрсдэлтэй аливаа асуудлыг шийдвэрлэх үе шатууд юм.



Гэрийн даалгавар:

1. Энэхүү системийн үндсэн өгөгдлийг олж, тайлбар хийнэ үү.

Бараа зарах автомат хувьд үндсэн өгөгдөл нь бараанууд, автомат машин мөнгө хэрэгтэй.

Автомат машины хувьд барааны нөөц бол төрөль хэмжээ,үнэ зэрэг шинж чанар болон шинэ ширхэгийн тоо хэмжээ бодох, өөрийн үнээ тодорхойлох зэрэг үйлдлээр үндсэн өгөгдөл болж байгаа. Харин автомат машин нь бараа гаргах, барааг нэмж дүүргэх, барааны нөөцыг хадгалах, борлуулалт хийхэд тусламжтай үндсэн өгөгдөл болж чадаж байна. Мөнгө бол автомат машин хэрэглэгч бараа 3ыг хооронд холбох үндсэн өгөгдлөл болж байна.

1. Энэ системийн үндсэн үйлдлийг олж,тайлбар хийнэ үү.

Бараа зарах автомат хувьд үндсэн үйлдлүүд бол бараа сонгох, мөнгө оруулах, барааг гаргах зэрэг үндсэн үйлдлүүд болж байгаа.

Хэрэглэгч болон автомат хувьд хоорондын үйл ажиллагаа илэрхийлж байгаа тул үндсэн үйл ажиллагаа болж байна.

1. •Объектыг олж, тайлбар хийнэ үү.

Бараа зарах автомат хувьд бараа бүтээгдэхүүн, мөнгө, хэрэглэгч, автомат, борлуулалт зэрэг болж байна. Бараа бүтээгдэхүүн бол өөрийн гэсэн барааны нэр, үнэ, тоо ширхэг хэмжээтэй, үнэ өгөх, шинэ ширхэг тоогоо гаргах зэрэг үйлдлүүдийг илэрхийлж байгаа. Мөнгө бол борлуулалт хйих зориулалттай бодит зүйл байгаа. Автомат барааг гаргах, оруулах, үлдэгдэл мөнгө өгөх, борлуулалтын тайлан гаргах бодит зүйл байгаа.

# Бүтцийн хөгжүүлэлт

Бүтцийн хөгжүүлэлтийн аргад функц хандлагат хуваалт болон өгөгдлийн хандлагат хуваалт хамаардаг. Бүтцийн хөгжүүлэлтийн үе шатанд бүтцийн шинжилгээ, зохиомжыг гаргадаг.

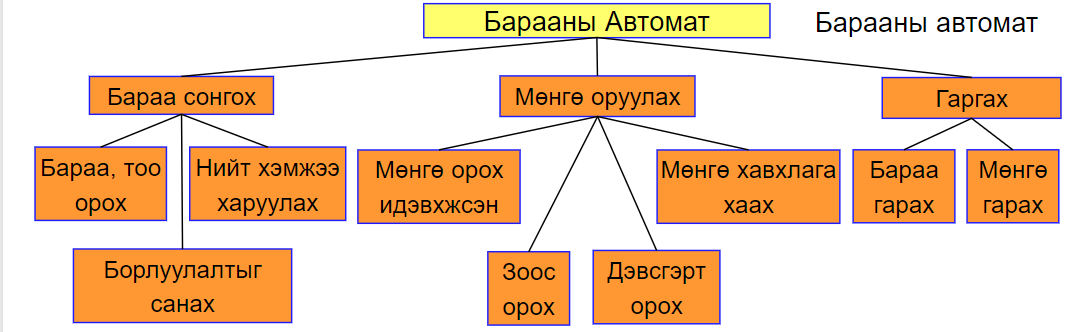
Бүтцийн шинжилгээ нь тухайн системийг өгөгдлийн урсгалын диаграммааар дүрсэлдэг.

Бүтцийн зохиомж үед бол функц болгоны бүтцийн диаграмм зурж, өгөгдөл хиисвэрлэлт хийдэг.

Сул тал бол хүний бодох аргачлалтай тохирдоггүй. Хөгжүүлэлтийн үе шат хооронд байнга өөрчлөгдөж, эвдрэл гэмтэл үүсгэдэг. Дахин ашиглах байдлыг дэмждэггүй.

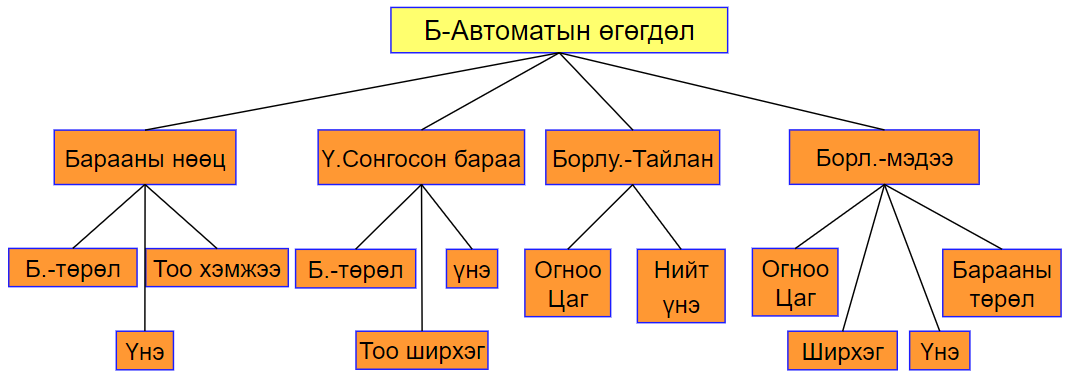
# Функц хандлагат хуваалт:

Тухайн хэрэглээний болон програм хангамжын бүтээгдэхүүний функц болон үйлдлүүдийг тайлбарлан бичдэг. Үр дүнд бол системийн үйл ажиллагааг нарийн жижиг бүхэл хэсэг болгон жижиглэж байдаг. Сул тал тухайн системийн өгөгдлүүдийг орхигдуулж байдаг.

**Лекцийн жишээ:**

# Бүтцэн хандлагат хуваалт:

Тухайн системийн өгөгдлийг 2 хувааж үздэг. Боловсруулагдах өгөгдөл болон хадгалагдах өгөгдөл гэж хуваан үзэн системээ хуваадаг. Сул тал тухайн системийн үйл ажиллагааг орхигдуулж байдаг.

Лекцийн жишээ:

# Объект хандлагат хөгжүүлэлт

Тухайн системд өгөгдөл болон функц хандлагат хуваалтыг тайлбарлаж бичихээс гадна тэдгээрийн хоорондын уялдаа холбоог дүрсэлж байдаг.

Давуу тал:

1. Бодит байдлын харилцын уялдаа хамаардыг илүү сайн ойлгож болно.
2. Илүү өргөтгөх боломжтой удамшлын ойлголтоор
3. Дахин ашиглахуйц байдалтай байдаг.
4. Бусад хөгжүүлэлтийн үйл явцад шилжихэд асуудал үүсдэггүй ойлгомжтой байдаг.

Бүтцийн болон Объект хандлагат хөгжүүлэлтийн ялгаанууд:



Лекцийн жишээ



Функц хандлагат хуваалт тухайн объектийн үйлдэл

Өгөгдөл хандлагат хуваалт тухайн объектийн шинж чанар

# Объект хандлагат аргууд:

Үүнийг судлахаас өмнө арга гэдэг ойлголтыг тодруулах хэрэгтэй.

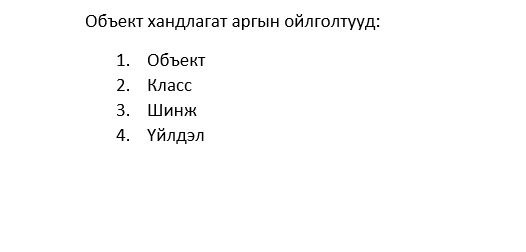
Арга бол аливаа зүйлийг хийхэд зориулагдсан системтэй үйл ажиллагаа юм. Арга бол дотроо үндсэн ойлголт, тэмдэглэгээ арга замаар тодорхойлдог.

Ойлголт бол Объект хандлагат аргад бол шинжилгээ, зохиомж, хэрэгжүүлэлт зэрэг процесст зайлшгүй байдаг.

Объект хандлагат аргын ойлголтууд:

1. Объект
2. Класс
3. Шинж
4. Үйлдэл
5. Удамшил
6. Зурвас
7. Холбоо
8. Ажлын явц
9. Багц, Сценари,Төлвийн автомат зэрэг ойлголтууд хэрэглэгддэг.

Тэмдэглэгээ бол 2 үндсэн хэсэгтэй.

 Текст График

Арга зам: Системтэй алхам болон системтэй дүрмүүд хэрэгтэй байдаг. Програм хангамжын бүтээгдэхүүний хийхэд шаардагдах процессуудын дараалалууд хамаардаг .

# **Лекц 3-4**

# Объект гэж юу вэ?

Объект гэдэг шинж чанар болон үйл хөдлөл бүхий юм объект гэнэ. Объектийн шинж үл өөрчлөгдөж байдаг. Харин шинжийн утга нь өөрчлөгдөж болно.

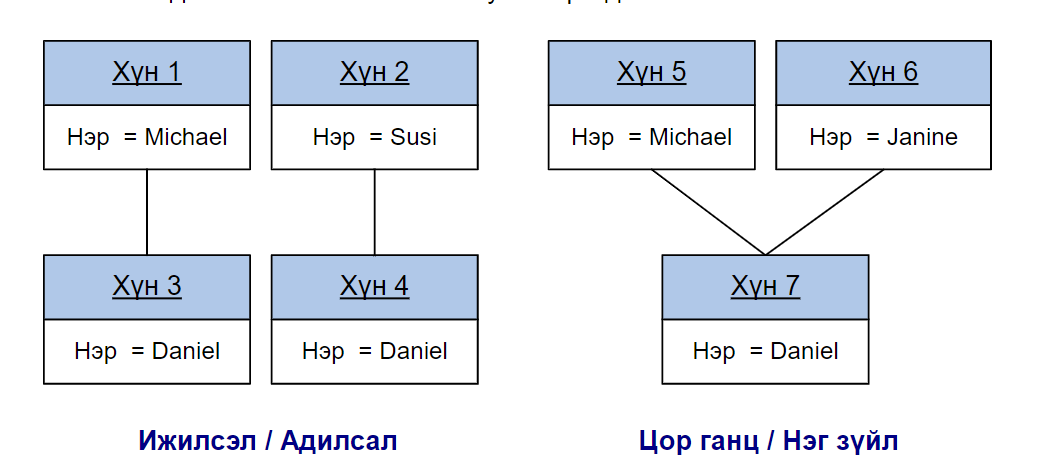
## Java Class and Objects - Easy Learning with Real-life Examples! - TechVidvan

## Объектийн онцлог

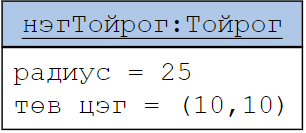
|  |
| --- |
| 1. Ямар нэгэн тодорхой төлөв(шинж тэмдэг) байна. |
| 1. Объектийг тодорхойлох зан ааш араншинтай байна. |
| 1. Бусад объектоос ялгарах цор ганц шинж чанартай байна. |
| 1. Өөр бусад объекттой холбоотой байж болно. |

Объектийн цор ганц байдал

Ямар нэгэн объект өөр бусад объектоос ялгардаг, өөр өөрийгөө өөрчилж чадахгүй , объектийн бусад онцлох шинжүүдтэй ямар нэг хамааралгүй байх ойлголт юм.

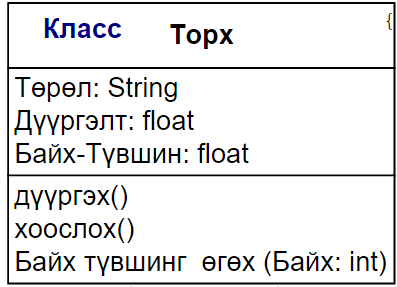


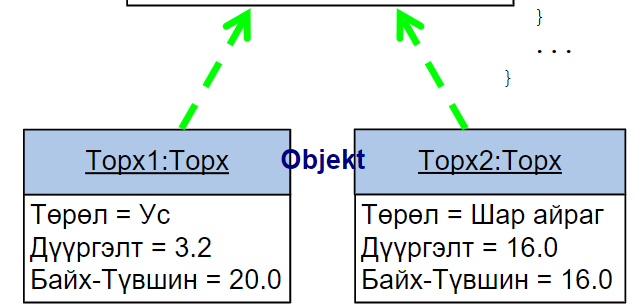
Объектийн загварчлалын нэгдсэн хэлэнд дараах байдлаар дүрсэлдэг.

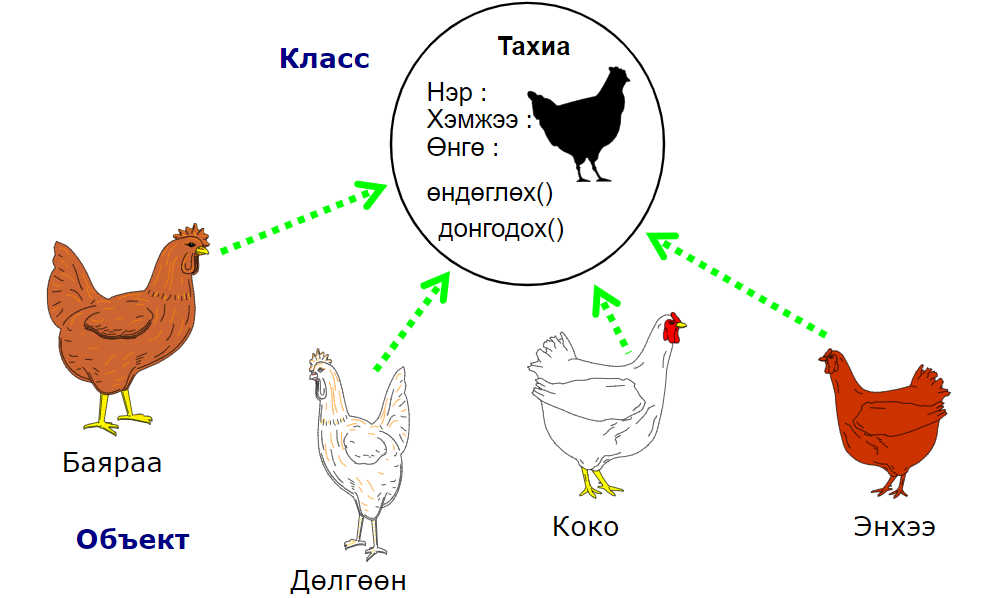
   
Класс гэж юу бэ?

Класс бол объектийг загварчлах зорилготой үүсмэл өгөгдлийн төрөл юм.

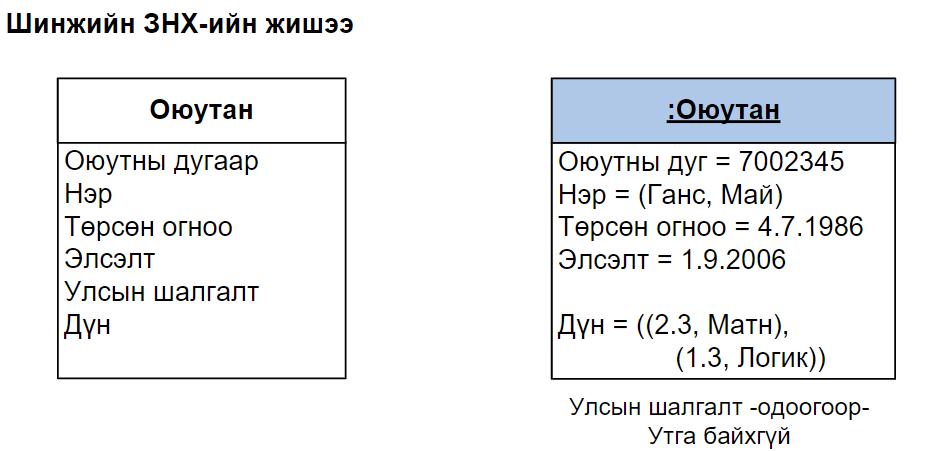
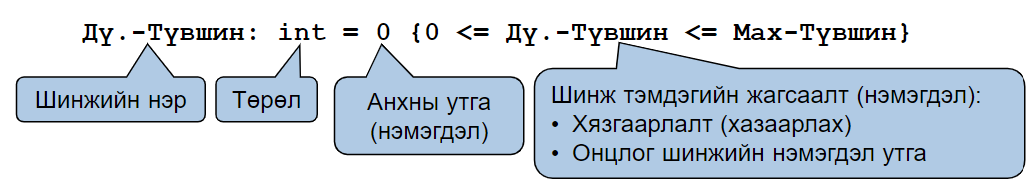
Классын загварчлалын нэгдсэн хэл дараах тэмдэглэгээг хэрэглэдэг.

Жишээ нь





Класс болон объектийн холбоо

* Класс объектийн шинж болог үйлдлийг тодорхойлдог.
*  Объектод шинжийн утгийг олгодог

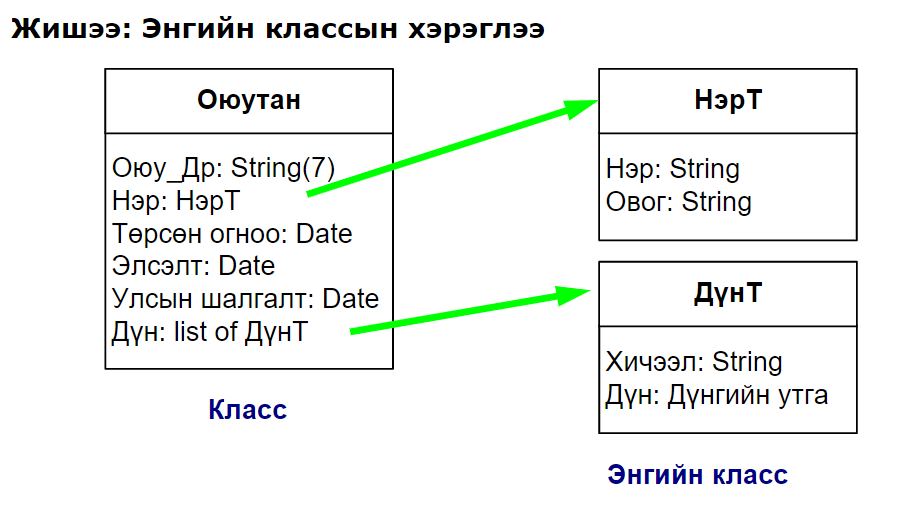
ЗНХ(загварчлалын нэгдсэн хэл) стандарт төрлүүд

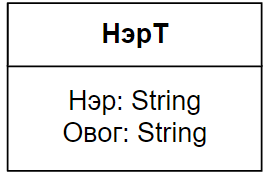
|  |  |
| --- | --- |
| Төрөл | Тайлбар |
| String | Тэмдэгтийн цуваа |
| Int | Бүхэл тоо |
| Float | 32 битийн хөвөх цэгтэй тоо |
| Double | 64 битийн хөвөх цэгтэй тоо |
| Fixed | Тогтмол цэгт тоо |
| Boolean | Үнэн худал илэрхийлэх |
| Date | Он сар |

Энгийн класс

Шинжийн төрлийг өөр классаар тайлбарлан бичсэн байдаг.

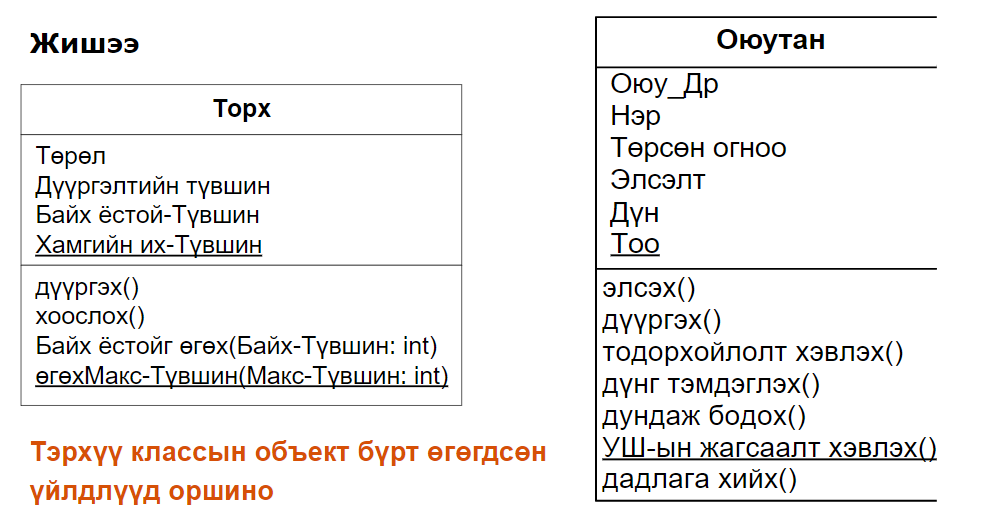
Энгийн класс ард Т угтвар авч байдаг.



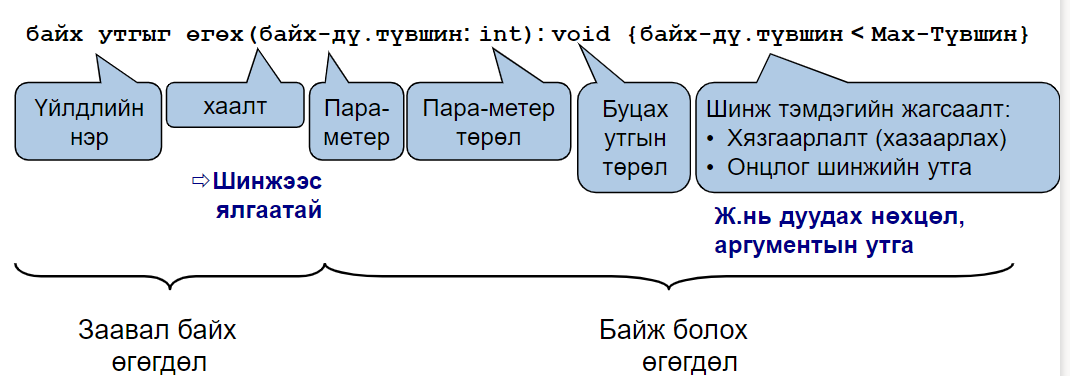


Классын шинж нь нэг классын бүх объектод шинжийн зөвхөн нэг л утга оршино. Шинжийн утгын өөрчлөлт тэрхүү классын бүх объеътод хамааралтай байдаг. Классын шинжийг доогуур зураасаар ялгаж өгдөг.

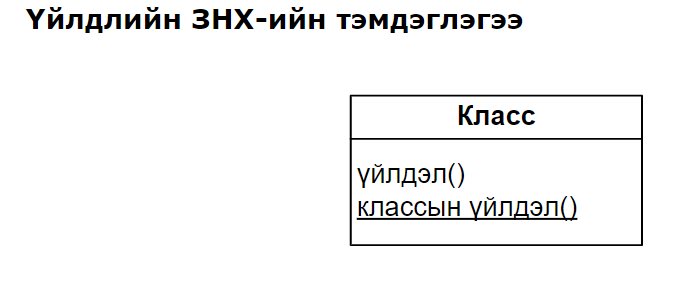
# Үйлдэл:

* Нэг классын бүх объект нэг ижил үйлдэлтэй байна.
* Үйлдэл бүр объектийн шинж бүрд шууд хандаж болдог.

Жишээ:



Классын үйлдлийг дараах ЗНХ дүрсэлдэг.



Үйлдэл бол юу гүйцэтгэж байгааг дүрсэлдэг, үйл үг байдаг. Класс.үйлдэл() гэж классын гаднаас хэрэглэж болно.

ЗНХ-ийн өргөтгөсөн механизм

* Стерео төрөл
* Шинж тэмдэг

Стерео төрөл:

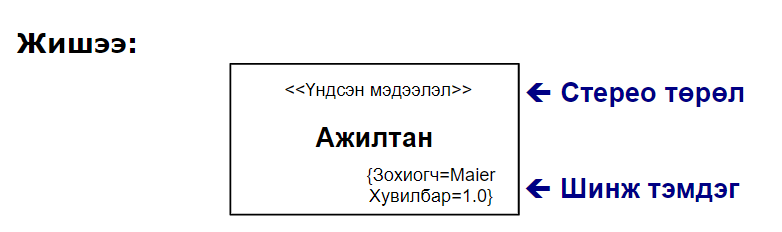
Классын үйлдэл болон шинж чанарыг нэр олгон ангилдаг.

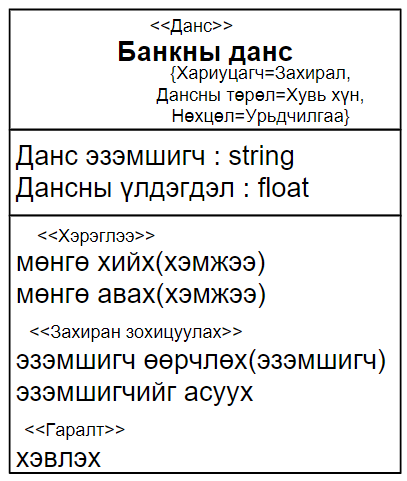
Классын хэрэглэж болохуйц зориулалтыг нэрлэхэд хэрэглэдэг.   
Стерео төрлөөр нийт класс мөн түүнчлэн шинж болон аргын бүлгийг хоёуланг нь тайлбарлан бичиж болно.

Шинж тэмдэг:

–Загварын тодорхой элементын онцлог шинжийг тайлбарлан бичдэг.

–Олон шинж тэмдэгийг ямар нэг жагсаалтад нэгтгэж болно:





# **Лекц 5- 6**

Статик болон динамикын талаар ойлголт

ОСтатик загвар бүтцийг харуулж байдаг. Харин динамик загвар ажиллагааны үйл явцыг харуулдаг.

|  |  |
| --- | --- |
| Статик загвар | Ойлголт |
| Класс | Объектийг загварчлах зорилготой үүсмэл өгөгдлийн төрөл. |
| Холбоос | Элементийн харилцааг тайлбарлан бичдэг. |
| Удамшил | Классын ерөнхийлөлтийг тайлбарладаг. |
| Шинж | Классын шинжийг тайлбарладаг. |
| Багц | Системийг дэд хэсгүүдэд хуваана. |
| Динамик загвар | Ойлголт |
| Ажлын явц | Гүйцэтгэж буй даалгаврыг хийсвэрлэлтийг маш нарийн түвшинд тайлбарладаг. |
| Сценарь | Объектууд хоорондын харилцааг харуулдаг. |
| Төлөвийн автомат | Төрөл бүрийн үзэгдэлд объект хариу үйлдэл үзүүлэх |

# Холбоос

## Нэгээс олон классын объект хоорондын хэлхээ холбоог загварчилдаг.

Холбоосын шинж чанар

* Өндөр эрэмбийн холбоос байдаг.
* Рефлексив холбоостой байж болно.
* Холбоос нь чиглэлтэй

Холбоосын ангилал

* Энгийн холбоос
* Бүрдмэл
* Нийлмэл

Загварчлалын нэгдсэн хэлний холбоосын тэмдэглэгээ

* Хоёртын холбоо

Нэг эсвэл хоёр классын хоорондох шулуун

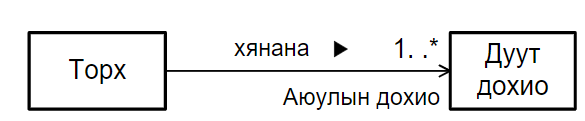
Холбоосын нэр

Шулууны төгсгөл бүрт харьцааны тоо байдаг

Төгсгөл бүрд үүргийн нэр байна.

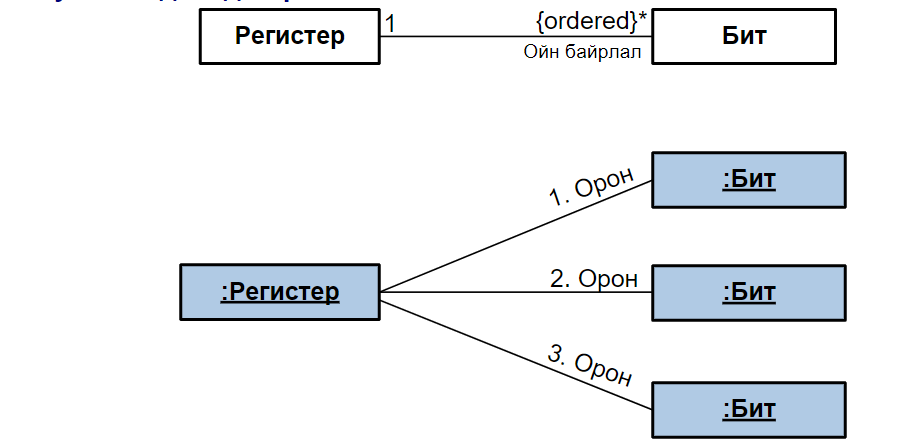


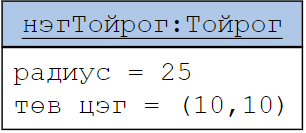
Чиглэлтэй холбоос

* Зөвхөн нэг объектод харьцааны тоог илэрхийлсэн байдаг. Сумаар тэмдэглэдэг.

Эрэмбэлэгдсэн холбоос

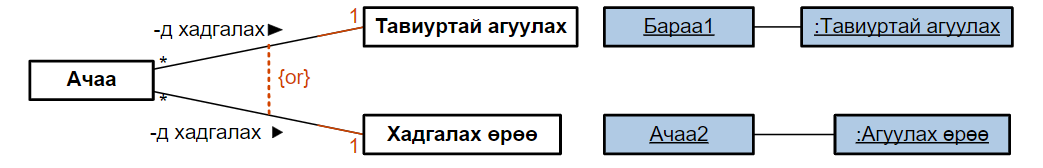
* Объектийн холбоосын олонлог эрэмблэгдсэн гэдгийг харуулна.
* Харьцааны тоо нэгээс дээш байх боломжтой
* Эрэмблэгдсэн гэсэн түлхүүр ашиглана.



   
Холбоосын хязгаарлалт

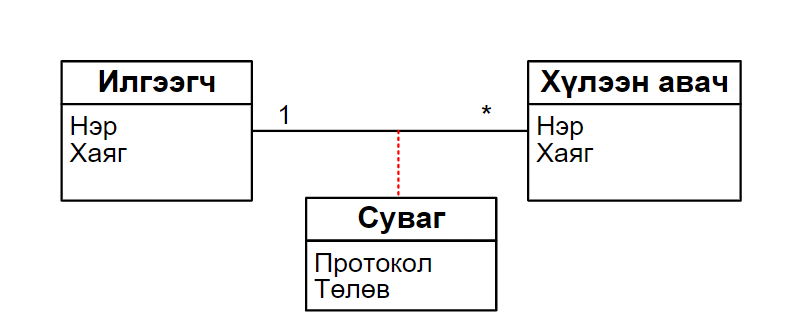
* Дандаа биелэгдэх ёстой нөхцөл
* Хязгаарлалт хийж болно.

Холбоосын хязгаарлалт- Or

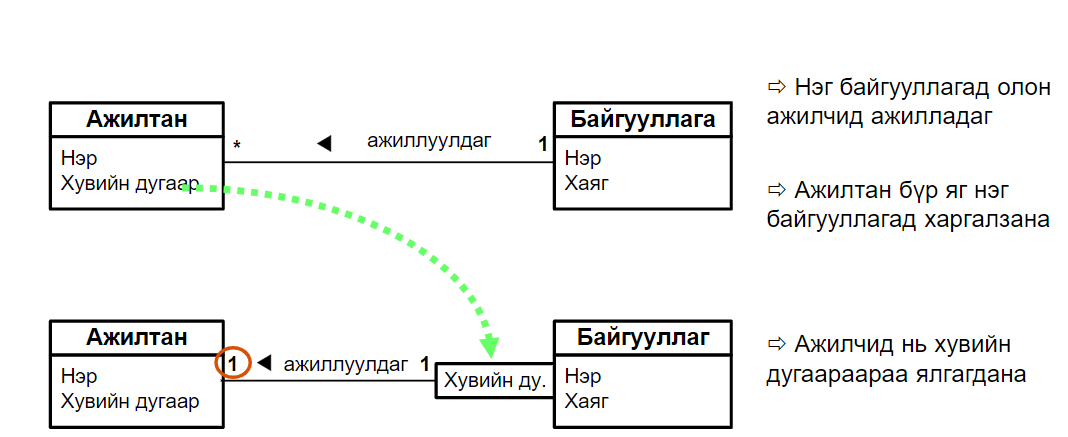
* Хугацааны дурын цэг бүрт боломжит олон холбоосын зөвхөн нэг нь хүчинтэй байна.
* Олон холбоосыг хамааруулж болно.

Холбоос класс

* Холбоос нь классын шинжтэй байж болно
* Классын тэмдэглэгээ бол тасархай шулуун
* Холбоос болон холбоос классын нэр нь үргэлж адилхан



Бүлэглэсэн холбоос

* Объектийн хооронд тусгай шинжээр нь хуваах, түүний утга нь эсрэг талдаа нэг эсвэл хэд хэдэн утгыг илэрхийлэх
* Бүлэглэсэн холбоосыг тэгш өнцөгтөөр дүрсэлдэг.

# Бүрдмэл холбоос

Классын хоорондын бүрдэл-бүхэл харьцаа юм. Бүрдэл холбоосын ангилалын нэг юм.

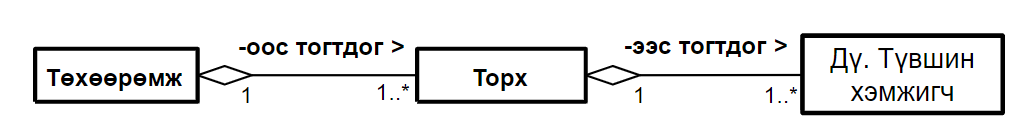
Бүрдмэлийн онцлог шинж:

* Тэгш хэмт бус

Бие биенээ бүрдэл байж болохгүй

* Дамжих чанар

Хэрэв тухайн классын бүрдэл хэсэг бол өөр нэг классын бүрдмэл болвол түүний бүрдмэл хэсэг түүний хэсэг болно.



|  |  |
| --- | --- |
| № | Бүрдмэл нь хэзээ өгөгдсөн байдаг вэ |
| 1 | Классын бүрдмэл хэсэг болж таарч байна уу |
| 2 | Бүхэл хэсэг болох зарим үйлдэл бүрэлдэхүүн хэсэгт таарч байнаа уу |
| 3 | Бүхэл хэсгийн шинж нь бүрэлдэхүүн хэсэгт таарч байна уу |
| 4 | Тэгш буст хэмээр илэрхийлэгдсэн байна уу? |

Нийлмэл холбоос

Бүрдмэл холбоос бодоход нягт холбоотой хэлхээтэй, оршихуйн хамааралтай бүрдэл холбоос юм.

Нийлмэлийн онцлог шинж

* Агуулагч классын цор ганц бүрэлдэхүүн хэсэг байна.
* Бүхлийг хуулбал, бүрдэл нь мөн хуулагдана.

Жишээ нь



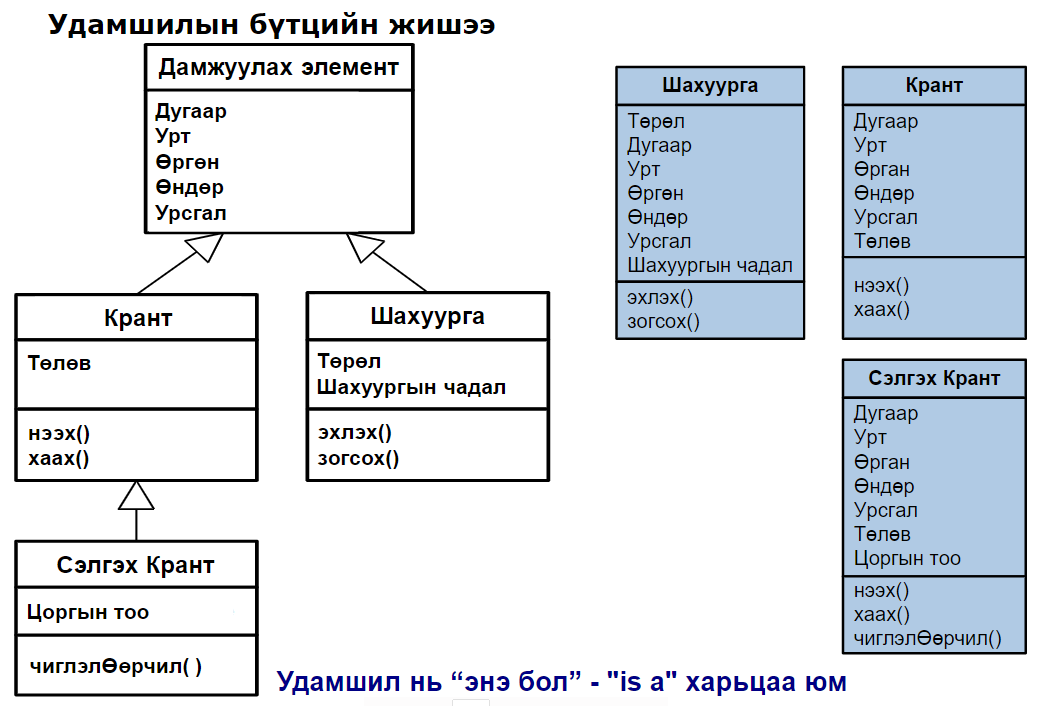
Удамшил

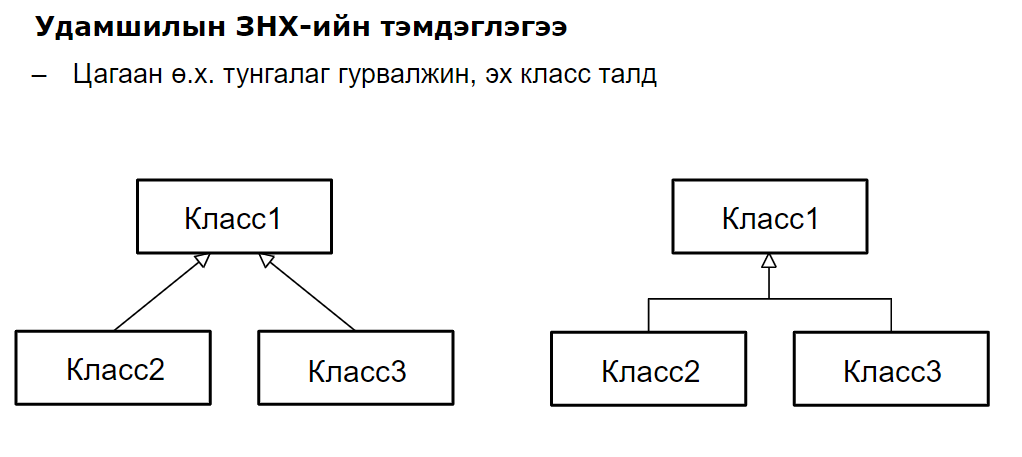
Ерөнхий класс болон түүнээс нарийвчлан задласан классын холбоо юм.

Удамшил шаталсан хэлбэртэй бүтцэд оруулах хийсвэрлэлтийн зарчим юм.

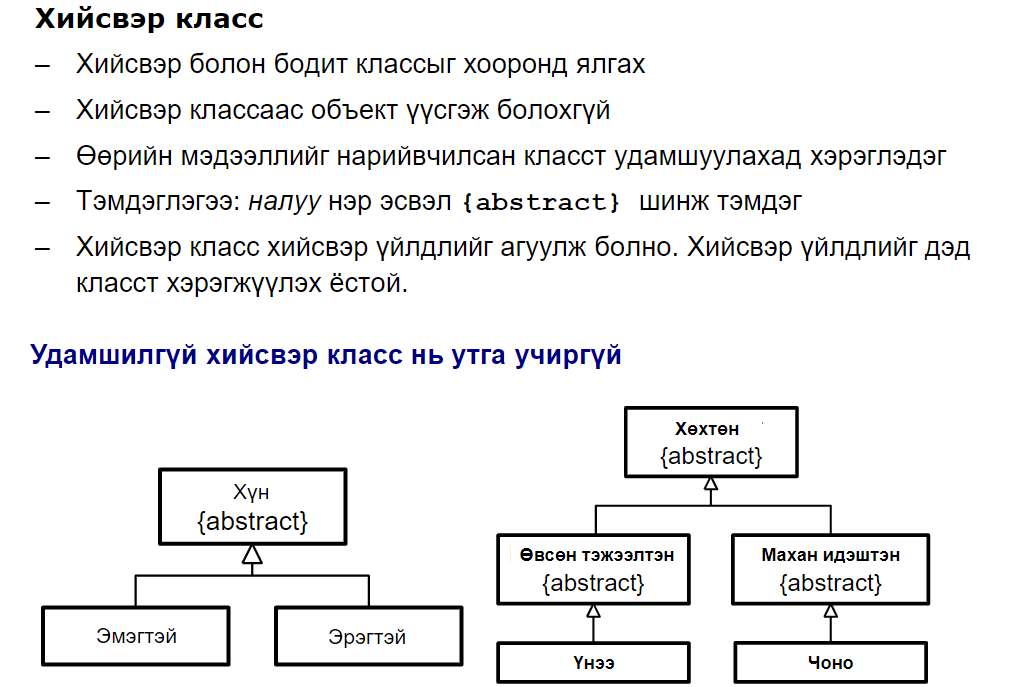
Ерөнхий класс- эх класс

Нарийвчласан класс – дэд класс





Удамшлаас үйлдэл болон шинж чанар удамшдаг.



# **Лекц 7-8**

Ажлын явцын талаарх ойлголт

Аливаа нэгэн зорилгод хүрэхийн тулд тоглогчидоос болон тэдгээрийн хоорондын харилцаа хамаарлаас бүрдэж хүссэн үр дүнгээ бий болгох аргийг ажлын явц гэнэ.

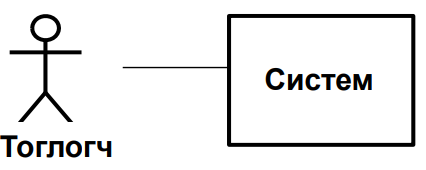
|  |  |
| --- | --- |
| Ажлын явц | Ажлын явцын нэр |
| Зорилго | Ажлын явцыг амжилттай гүйцэтгэх үеийн ерөнхий зорилго |
| Ангилал | Анхдагч, Хоёрдогч, сонголт гэсэн ангилалттай. |
| Угтвар нөхцөл | Ажлын явц эхлэхийн өмнө хүлээгдэж байгаа төлөв |
| Тайлбарлалт | Эхний ажиллагаа болон хоёр дахь ажиллагааг тайлбарлан бичдэг. |
| Дараах нөхцөл амжилттай | . |
| Дараах нөхцөл амжилтгүй |  |
| Тоглогч | Ажлын явцыг гүйцэтгэх тоглогч |
| Өргөтгөл | Эхний ажиллагааны цар хүрээний өргөтгөл |
| Хувилбар | Эхний ажиллагааны хувилбарын гүйцэтгэл |

# Ажлын явцын зорилго:

* Хэрэгжүүлж байгаа програм хангамжийн шаардлагад ажлын явцын задаргааг хийх
* Системийг хүссэн үр дүнд хүрэхийн тулд ямар даалгаврыг шинэ системээр хийж дуусгахыг олж тогтоох
* Шинжилгээний үеийн асуудлыг зохион байгуулах алхамыг агуулсан байнаа гэдгийн урьдчилан хэлэх хэцүү байлаг учир нарийн задлаж байдаг.

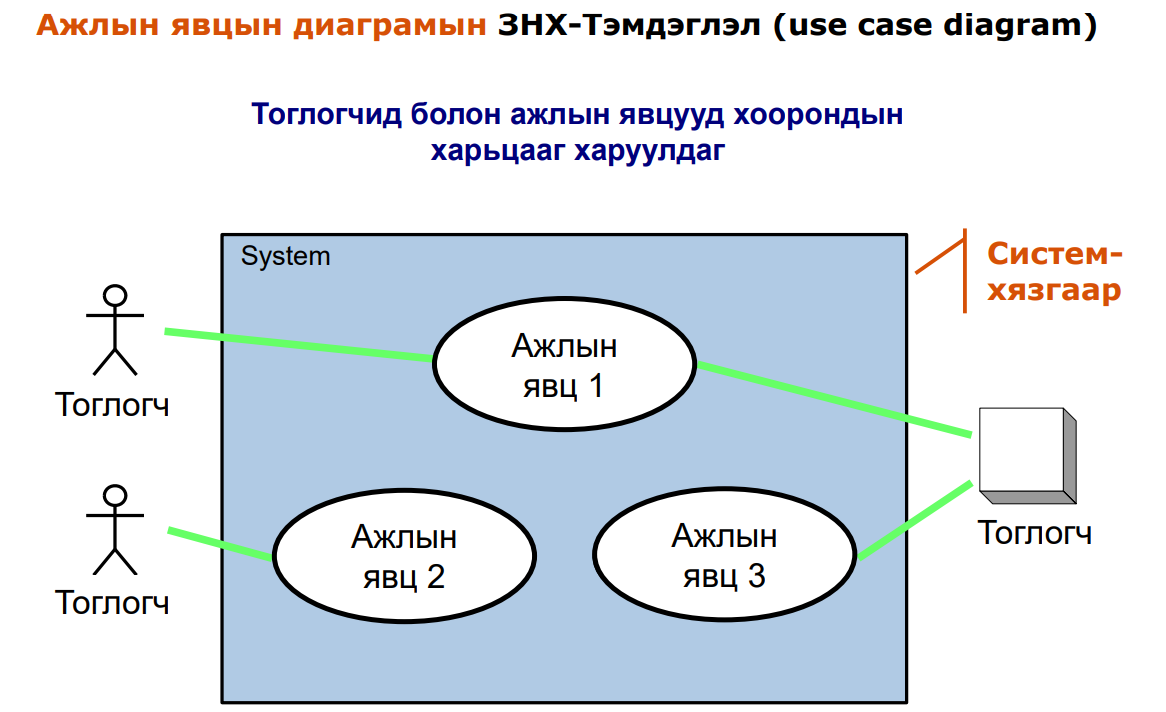
Тоглогч Системийн хэрэглэгчийн тоглож буй үүрэг

Тэмдэглэгээ Жишээ



Ажлын явцын хэрхэн тайлбарлаж бичдэг бэ?

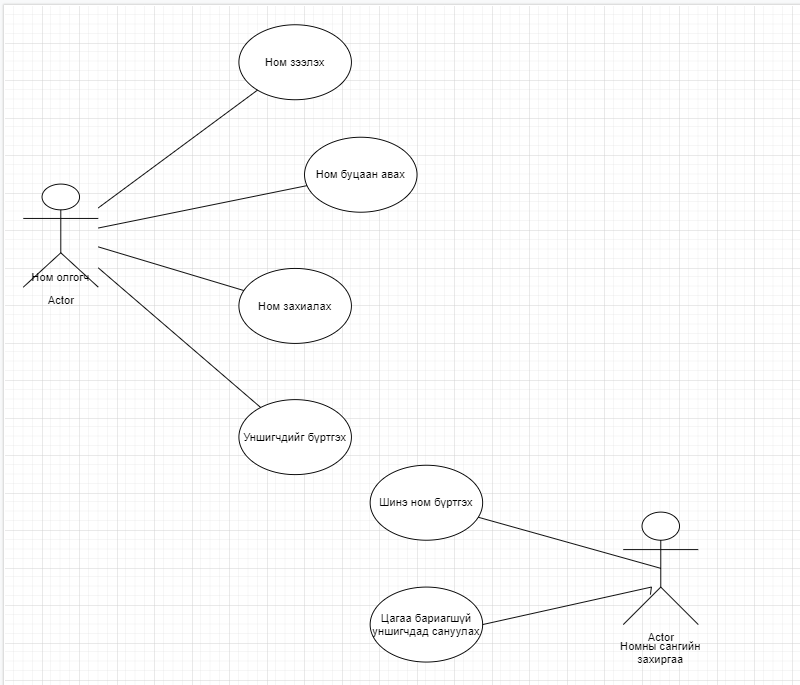
* Ажлын явцыг формал болог хагас формал хэлбэрээр тайлбарлан бичдэг
* Тус тусд нарийн задлаж дараалал хэлбэрээр тайлбарлан бичдэг.
* Ойлгомжтой байдлыг илүү болохын тулд тайлбарлана.
* Задаргааны хэв загварыг хэрэглэх
* Үргэлж хэрэглэгчийн хэсгээс хамаарлгүй хөгжүүлэгч мөн оролцож тайлбарлан бичдэг.



Ажлын явцын жишээ (Лекцийн торхны жишээ)



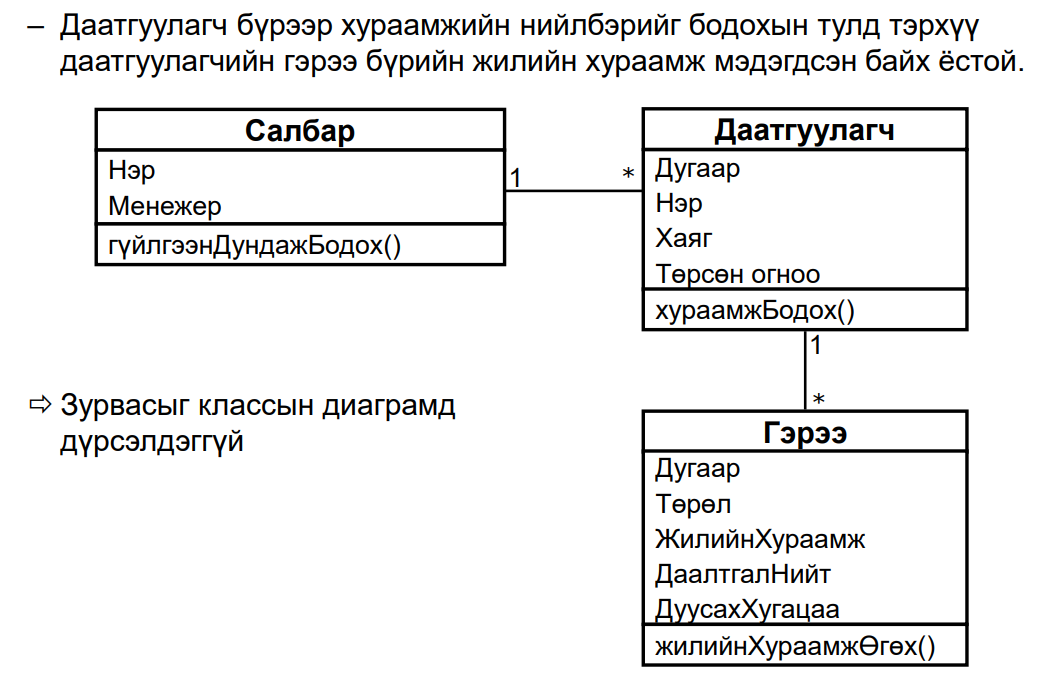
Дасгал ажил Use case



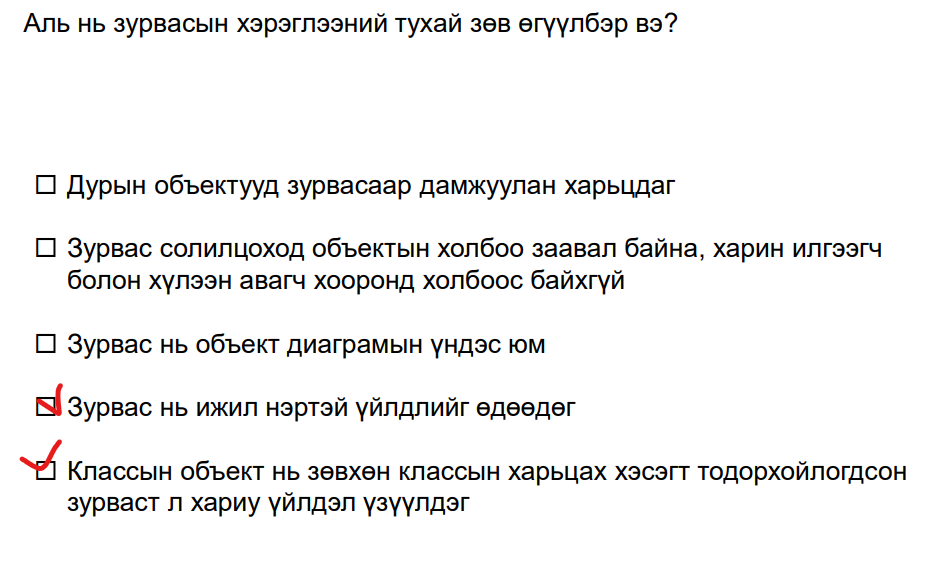
# ЗУРВАС

## Захиалагч болон хэрэглэгчийн хүссэн үйлчилгээг хүлээн авагч руу явуулж буй шаардлага юм.

* Хүлээн авагч тухайн зурвасыг тайлаад, үйлдлийг хэрэгжүүлдэг.
* Зурвасыг загварчлалын нэгдсэн хэлийн янз бүхийн диаграммд хэрэглэдэг.

Лекцийн жишээ 1

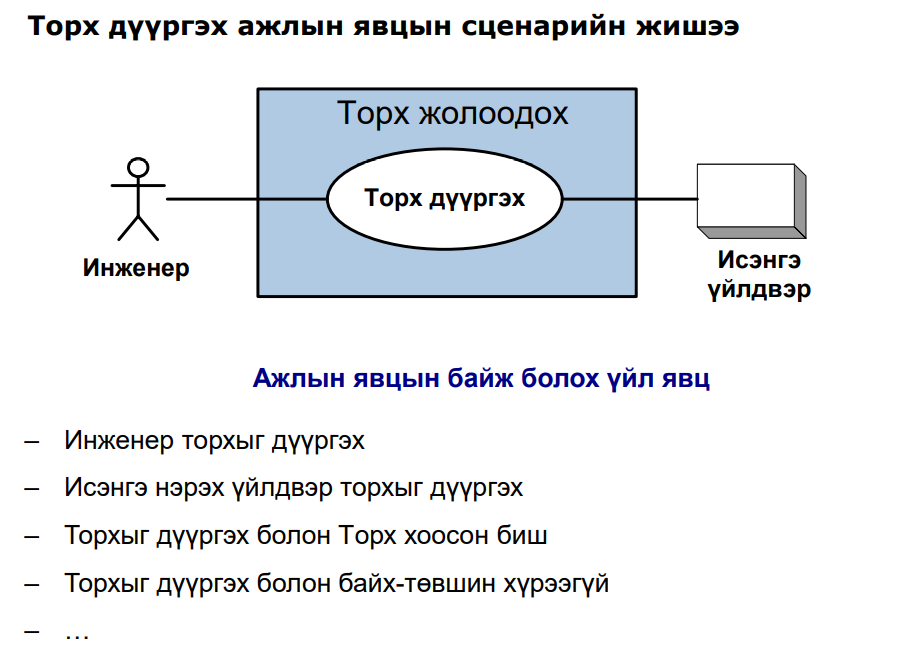
Асуулт1

b

# Сценарь

Тодорхой нөхцөл байдалд гүйцэтгэгдэж боловсруулалтын дэс дарааллын алхамууд юм. Сценарь хоёр ангилал ажлын явц амжиллтай боловсруулалтын тайлбарладаг. Харин ажил явцын боловсруулалт нь бүтэлгүйтэх хүртэл байна.

Лекцийн жишээ:



Сценарь харилцан үйлчлэлийн диаграммаар дүрсэлэгддэг.

# Төлөвийн автомат:

Төлөв болон төлөвийн шилжилтээс тогтдог.

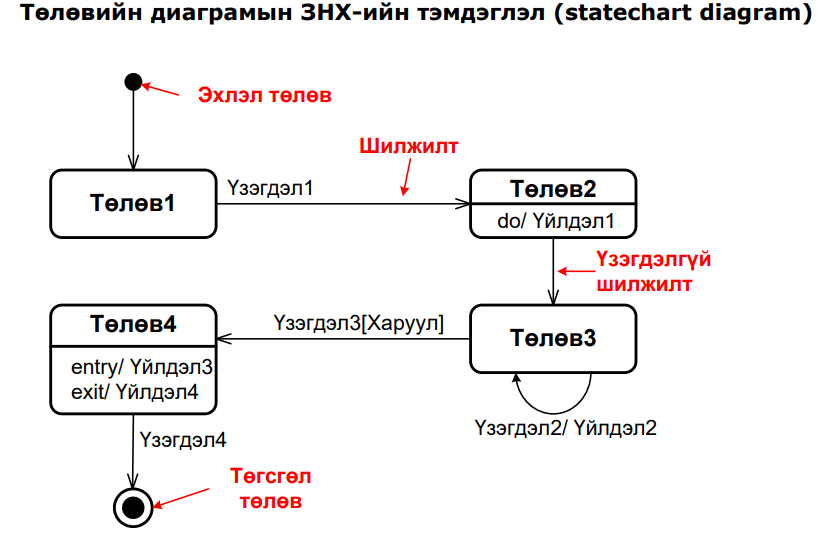
Төлөв гэдэг классын объект нь үзэгдлийг хүлээж буй цаг хугацааны интервал юм.

Төлөвийн шилжилт үзэгдлээр өдөөгддөг.

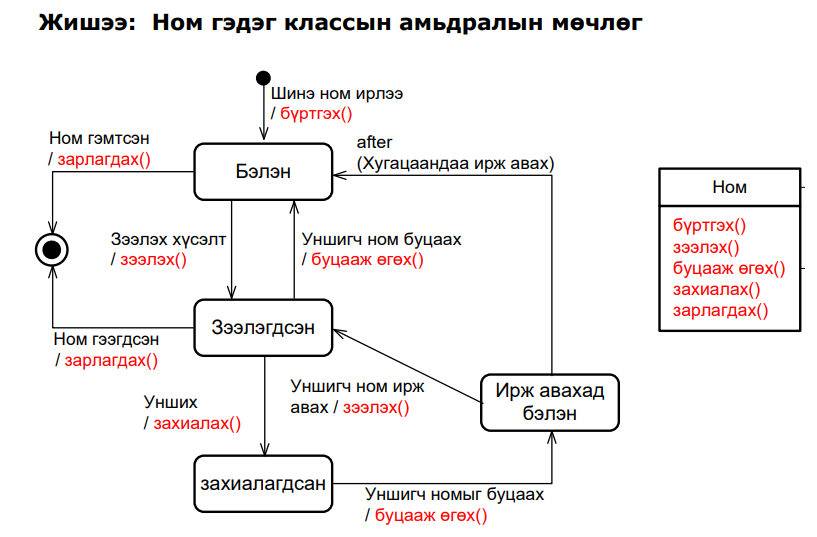
Төлөвийн автомат төлөвийн диаграммаар дүрсэлж бичдэг.

Төлөвийн автомат бол объектын амьдралын мөчлөгийг тайлбарлан бичдэг.

Төлөвийн автоматын тэмдэглээ:



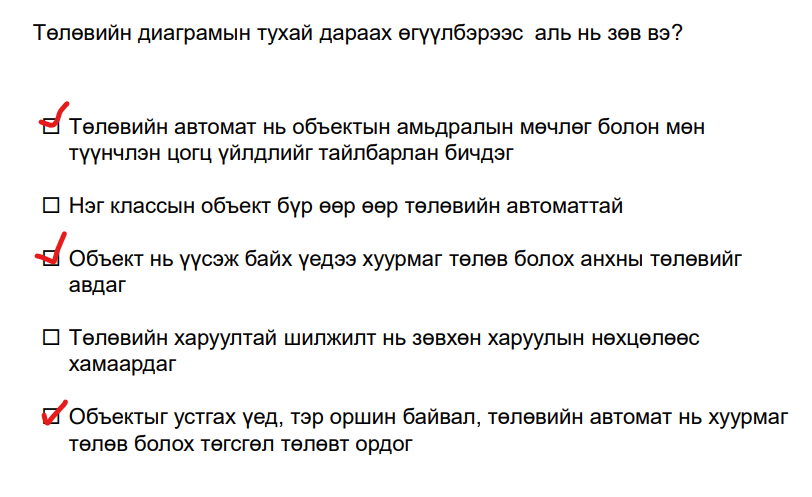
Лекцийн жишээ:



Эхлэлийн төлөв

Төгсгөлийн төлөв

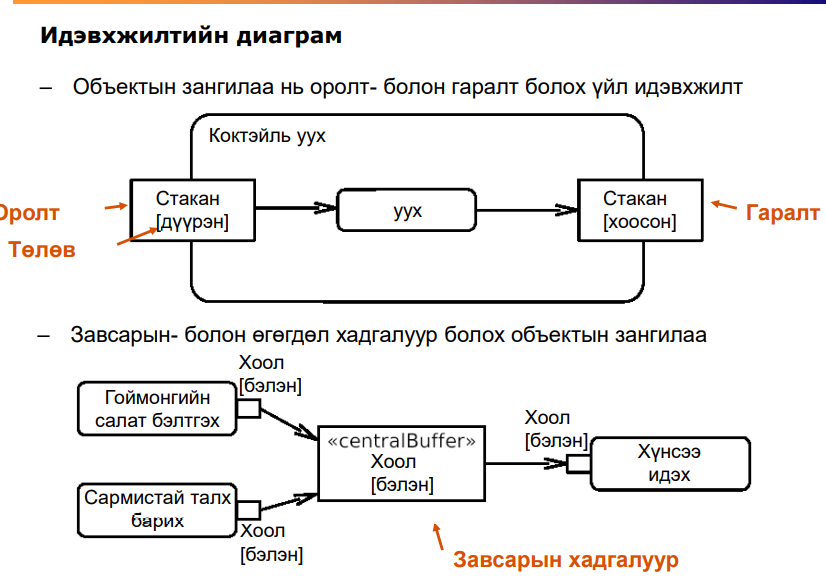
Төлөв



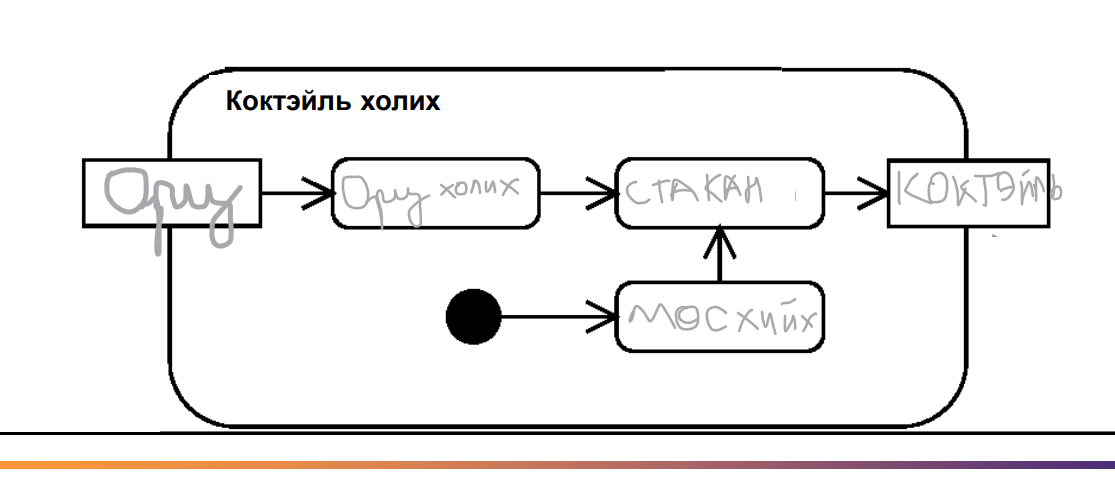
Үйл идэвхжилт:

Ажлын явцын задаргааг маш сайн хийхэд тохиромжой байдаг. Хяналтын болон объектийн зангилаа гэж ангилдаг. Хяналтын зангилаа нь шийдвэр гаргалт болн нэгтгэн нийлүүлтийг гүйцэтгэдэг. Объект зангилаа нь үйл өгөгдлийг дараагийнх руу нь дамжуулж өгдөг.

Үйлдэл гэдэг нэгж үйлдлийн идэвхжилтийн хамгйин бага нэгж юм.

Лекцийн материалаас авсан үйл идэвхжилтийн диаграмм

Асуулт үйл идэхжилийн диаграм

****

# **Лекц 9- 10**

# Шинжилгээний процесс

Шинжилгээний үеийн системтэй аргачлал

Хэрэгжүүлэлт

Загвар

Шинжилгээ\зохиомж

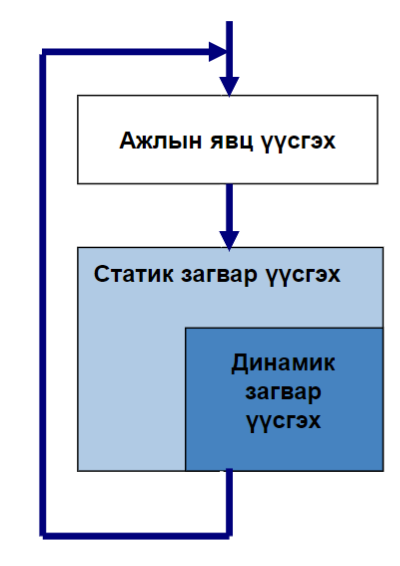
Асуудлын тодорхойлолтууд

ПХ систем

|  |
| --- |
| Шинжилгээ\зохиомж |
| Хуваалт болон хийсвэрлэлт  Жижиг, боловсруулж болох дэд хэсгүүдэд хуваах арга юм. |
| Бүтэцлэлт  Системийн дэд хэсгүүдийг тохиромжтой эрэмбэлэлт, систем хоорондын харилцааг олох |
|  |

Загвар үүсгэх аргачлал байж болох алдаанууд

* Динамик болон статик загварчлалыг хэтэрхий төвилгэх нь аюултай байдаг. Үргэлж өгөгдлийн загвар болон функцианоль бүтцийг объект хандлагат шууд хөрвүүлэх хэцүү байдаг.

Системтэй алхамыг тодорхойлох

Гол нь статик болон динамик загварыг тэнцвэржилтийг авч байдаг.

Макро процесс 2

* Өөрийн шаардлагаас чухал гэсэн ажлыг явцыг олж илрүүлэх
* Классыг олж илрүүлэх
* Статик загварыг үүсгэх
* Динамик загварыг үүсгэх
* 2 загварын харилцан нөлөөллийг авч үзэх

Макро процессын үйл явц:

1. Ажлын явцыг томъёолох

Өөрийн шаардлагаас ажлын явцыг олж илрүүлж, ажлын явцыг дүрслэх

1. Дэд систем үүсгэх

Том хэмжээний системийн хувьд дэд системүүдэд бүлэглэх

1. Классын диаграммаар байгуулах

* Классын диаграммыг олж илрүүлэх
* Холбоос, шинж болон шаардлагатай удамшлын бүтцийг олж илрүүлэх
* Холбоосыг бүрэн ойлгох
* Шинжийн задаргааг хийх

1. Сценарь хийх

Ажлын явцыг олон тооны нарийн задлаж сценараар тайлбарлах

1. Төлөвийн диаграмм

Класс бүрээр классын диаграммыг байгуулах

Лекцийн асуулт:



Хариулт:

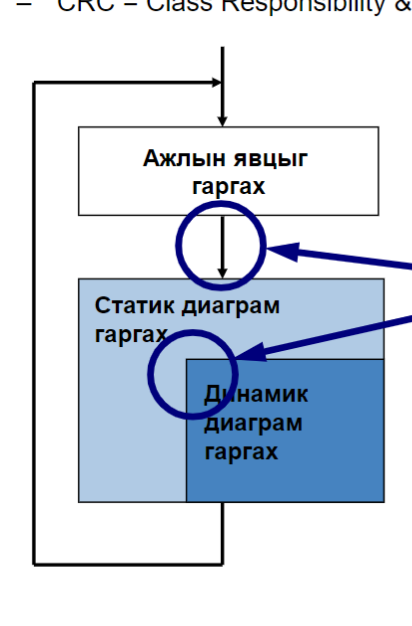
Аргачлал: Өгөгдөлд суурилсан макро процесс яагаад гэвэл хуучин өгөгдлүүд байна. Их хэмжээний өгөгдлийн загвар байгаа хуучин хэрэглэгчтэй харьцаж болох, ажиллаж буй систем зэрэг хуучин загвар их учир энэхүү аргачлалыг сонгож байна.

Ажлын явцыг

Хуучин систем, хэрэглэгч, ажиллаж буй систем, хэрэглэгчийн гарын авлага, файлын баримт бичигт тулгуурлаж үүсгэнэ.

Классын диаграммын гаргаж авах суурь ажлын явцын диаграмм сценарь хийх нь үндсэн суурь юм.

# Шинжилгээний процессийн КҮК карт



Шижилгээний процесст туслах зорилготой, классыг олж илрүүлэх, класс хоорондын холбоос болон классын диаграммыг байгуулах маш түлхэц өгөх арга юм.

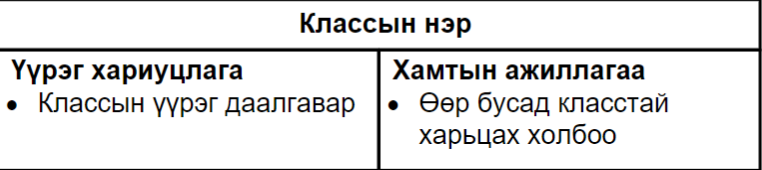
Классын нэртэй карт, энэхүү карт классын үүрэг болон шинж чанар мөн өөр ямар классуудтай холбогдох харьцааг харуулж байдаг.

|  |  |
| --- | --- |
| Ажлын явц | Ажлын явцын нэр |
| Зорилго | Ажлын явцыг амжилттай гүйцэтгэх үеийн ерөнхий зорилго |
| Ангилал | Анхдагч, Хоёрдогч, сонголт гэсэн ангилалттай. |
| Угтвар нөхцөл | Ажлын явц эхлэхийн өмнө хүлээгдэж байгаа төлөв |
| Тайлбарлалт | Эхний ажиллагаа болон хоёр дахь ажиллагааг тайлбарлан бичдэг. |
| Дараах нөхцөл амжилттай | . |
| Дараах нөхцөл амжилтгүй |  |
| Тоглогч | Ажлын явцыг гүйцэтгэх тоглогч |
| Өргөтгөл | Эхний ажиллагааны цар хүрээний өргөтгөл |
| Хувилбар | Эхний ажиллагааны хувилбарын гүйцэтгэл |

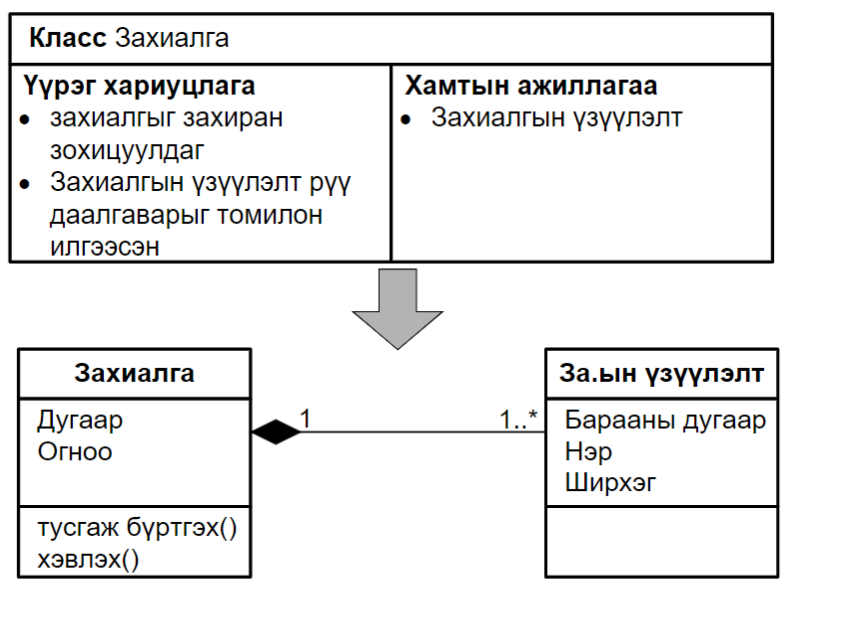
# Ажлын явцын зорилго:

* Классын үүрэг хариуцлагын олж илрүүлэх
* Холбоосын чиглэлийг олж илрүүлэх
* Класс болон тэдгээрийн холбоосын олж илрүүлэх

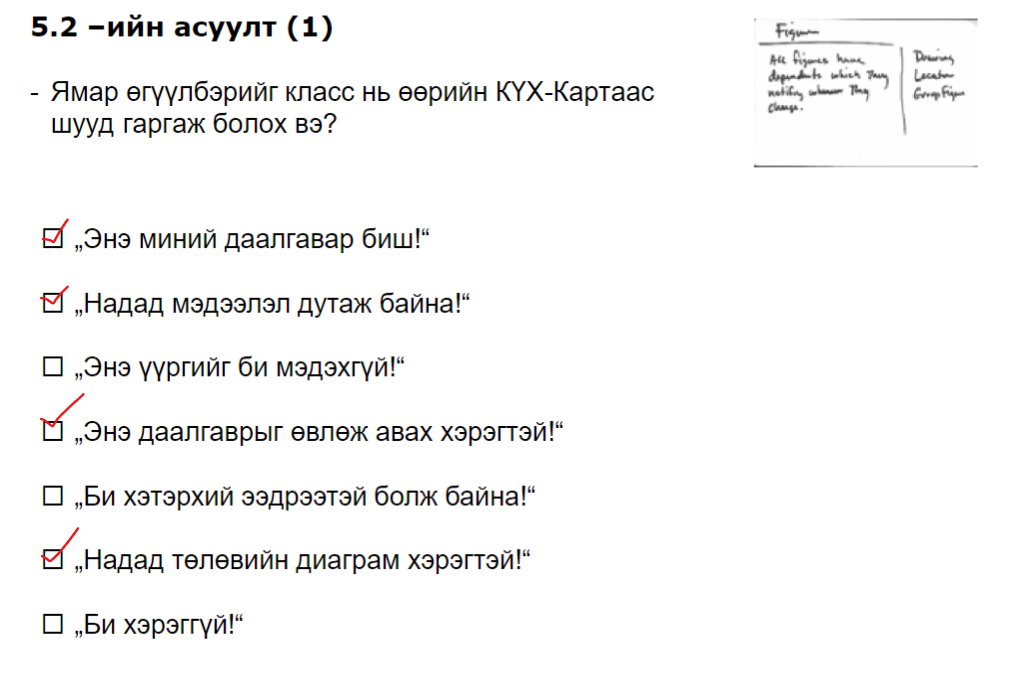
Дүрслэл:



Лекцний жишээ:



Дасгал:



|  |  |
| --- | --- |
| Класс Багийн уулзалт | |
| Үүрэг хариуцлага   1. Яг товлосон цагийг тодорхойлох 2. Уулзалтын өрөөг захиалах | Уулзалтын өрөө |

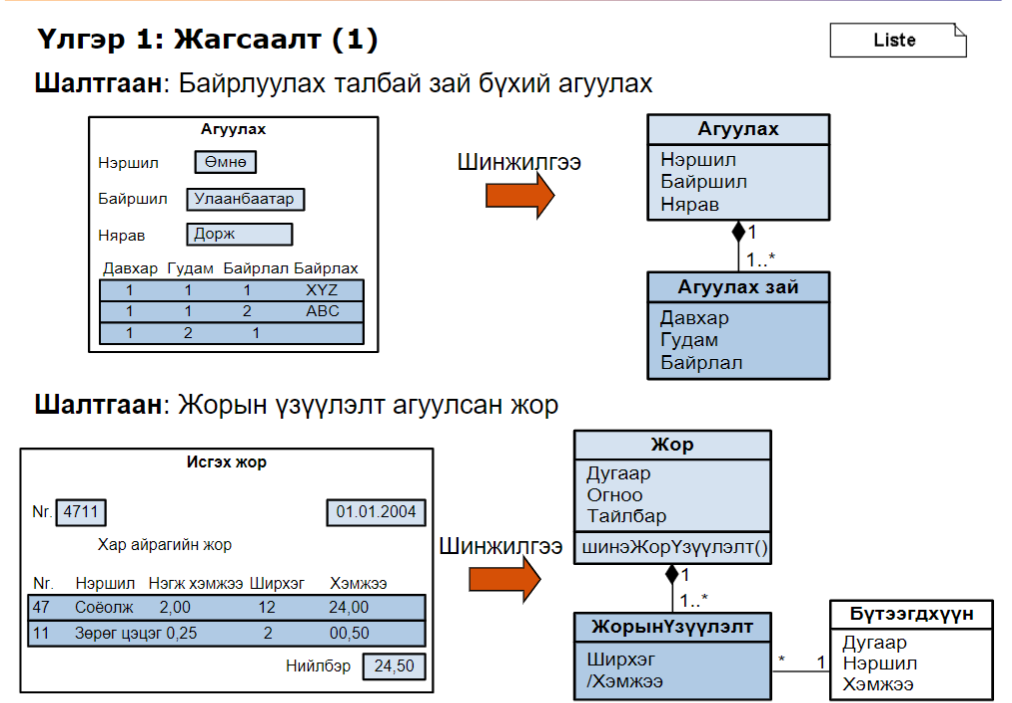
Шинжилгээний үлгэр загвар

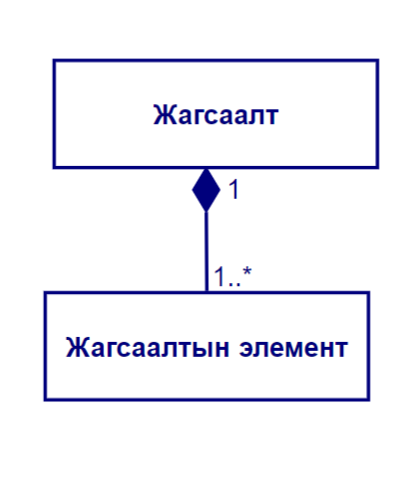
Тогтсон үүрэг харицлага болон харилцан үйл ажиллагаатай классуудын бүлэг юм .

Үлгэрийн хэрхэн тайлбарлаж бичих бэ?

1. Үлгэр бүр тодорхой нэртэй байдаг.
2. Үлгэрийн учир шалтгааныг тайлбарлаж бичих
3. Үлгэрийн шинж чанарыг олж таних
4. Нэг болон хэд хэдэн жишээгээр тайлбарлах

Лекцний жишээ:



Ерөнхий дүслэл:

Хяналтын жагсаалт

Шинжилгээний үед хэрэгтэй дүрэм болон чиг журмыг агуулдаг.

Хяналтын жагсаалтын төрөл

* Ажлын явц
* Дэд систем байгуулах
* Төлөвийн автомат
* Үйлдэл
* Класс
* Холбоос
* Шинж
* Удамшил

Ажлын явцыг үүсгэхэд зориулсан хяналтын жагсаалт

Ерөнхий дүрэм

Системийн шинжилгээ хийж байх үед анхдагч ажлын явцад анхаардаг.

Том хэмжээний системийг дэд системүүдэд хуваах

Ажлын явцын хяналтын жагсаалт:

|  |  |
| --- | --- |
| Бүтээлч алхам | 1. Тоглогч (actor) олж тогтоох |
|  | 1. Стандарт боловсруулалтын ажлын явцыг олж тогтоох |
|  | 1. Онцгой байдлын ажлын явцыг томьёолох |
|  | 1. Төвөгтэй ажлын явцыг жижиглэх |
|  | 1. Ажлын явцын хамтын хэсгүүдийг олж тогтоох |

Жишээ угаан хатаагч

Угаалгын машины үндсэн үйл ажиллагаа:

1. Угаах

Хүрдэнд усаа хийх

Угаалгын нунтаг хийх

Хүрдийг эргүүлэх

Хүрднээс ус гаргах

1. Булхах

Угаалгы хийхдээ угаалгын нунтаг ашиглахгүй хүйтэн усаар угаах

1. Сэгсрэх

Хатаах эргүүлийг хурдан эргүүлэх

1. Хатаах

Хүрдний эргэлт удаан

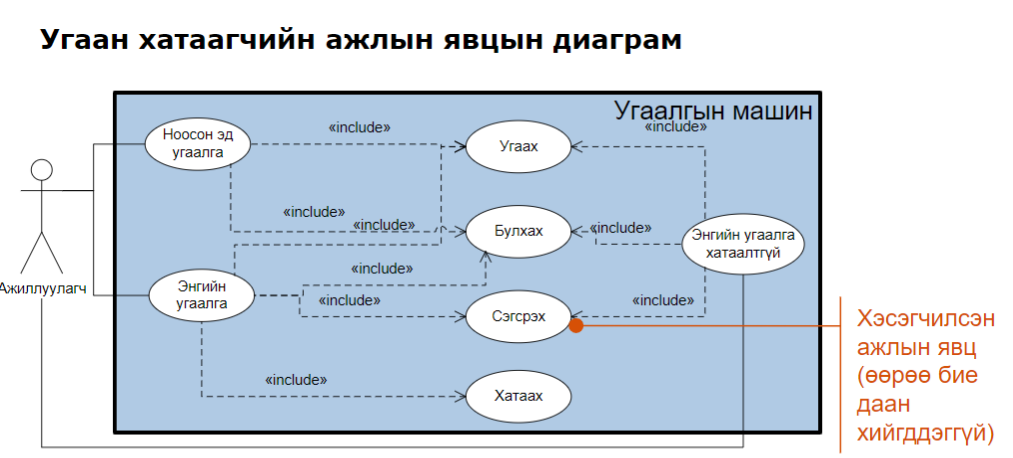


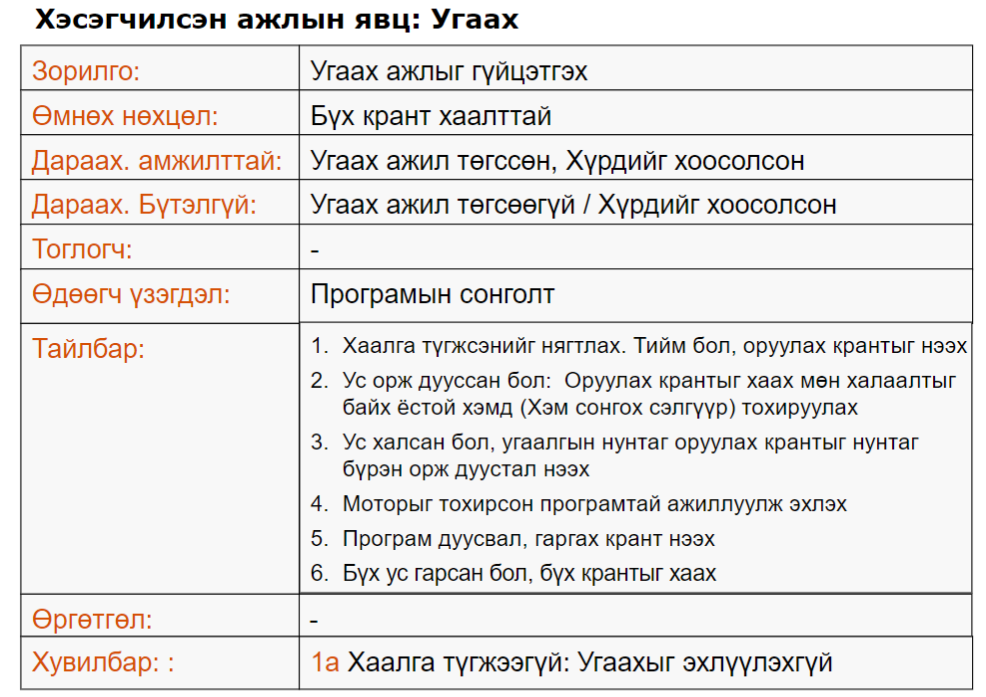
Ажлын явц олж тогтоосон байдал

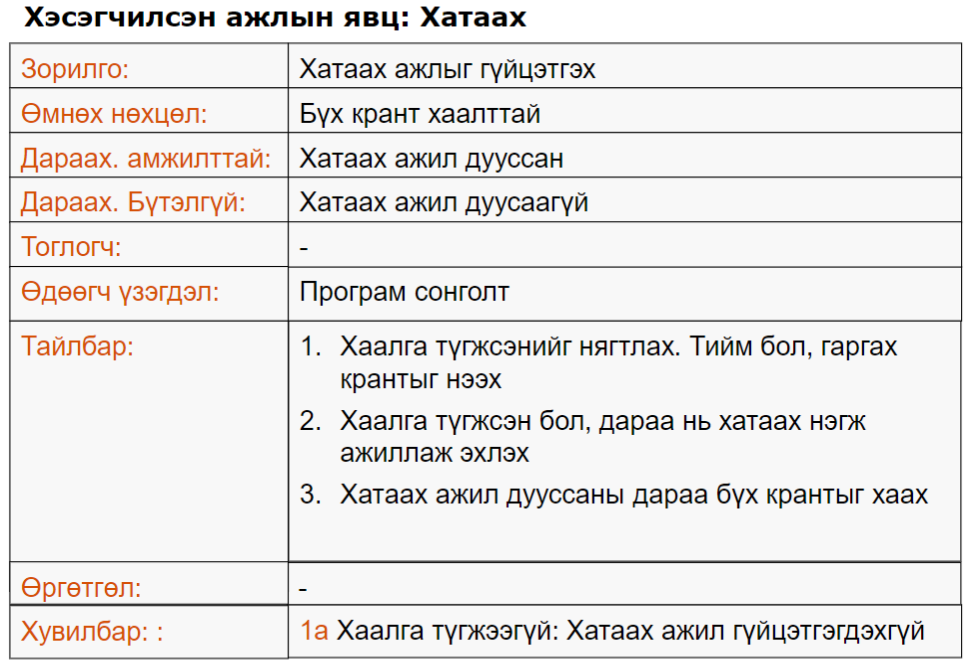
Энгийн угаалга хийх

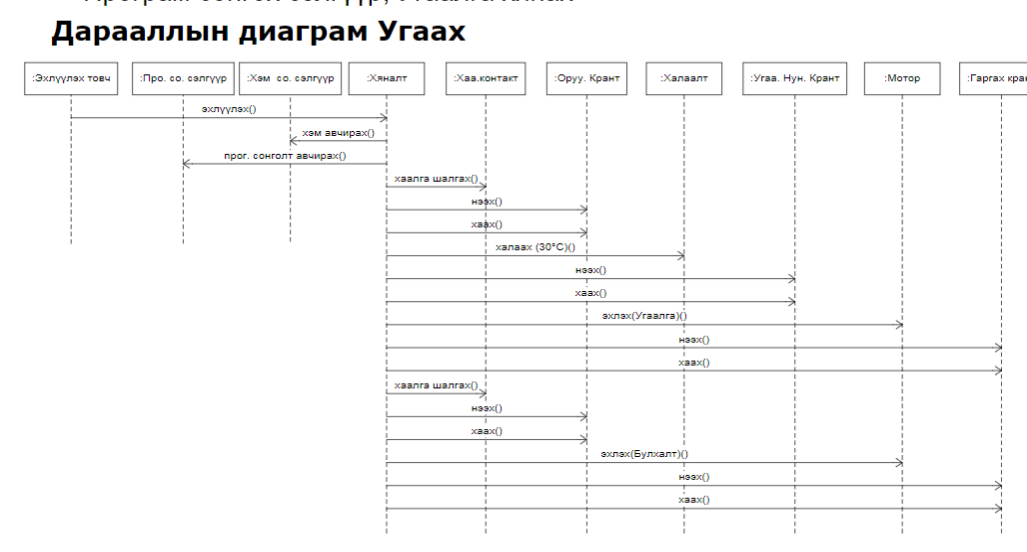
Энгийн угаалга хийх, хатаалтгүй

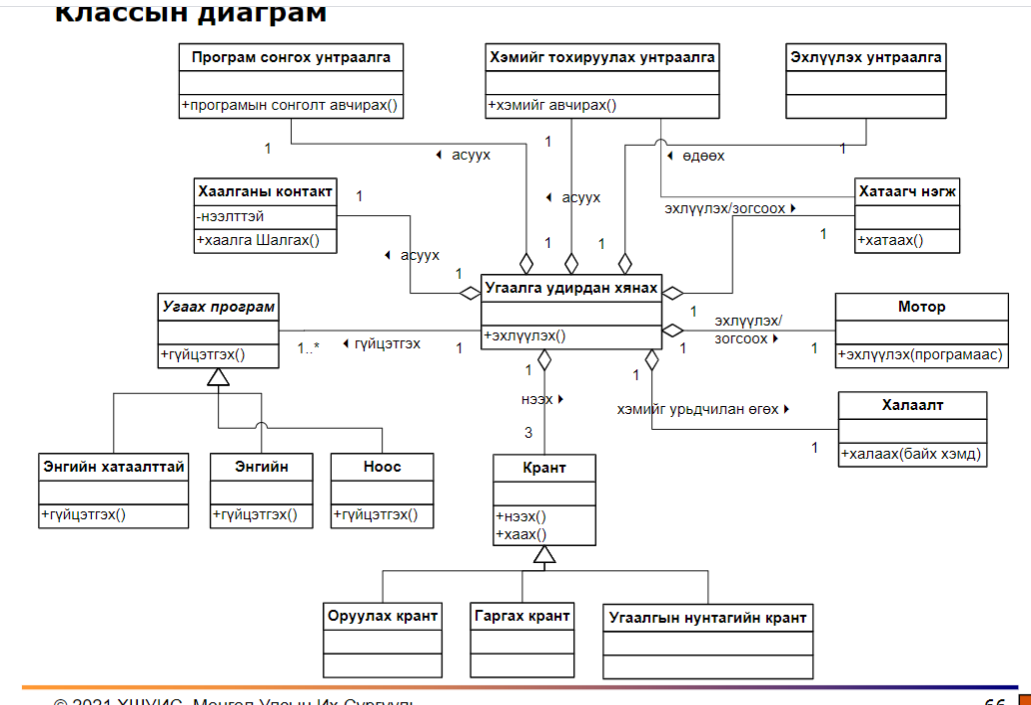
Ноосон эд угаалга хийх











# **Лекц 11- 12**

Объект хандлагат зохиомж ба шинжилгээний үе хоорондоо ялгаатай байдаг.

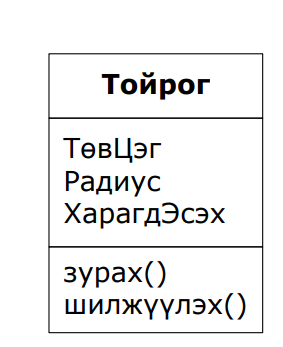
|  |  |
| --- | --- |
| Шинжилгээ | Зохиомж |
| Бизнесийн шийдэл | Техникийн шийдэл |
| Хэрэглэгчийн талаас харсан асуудлын талбар | Хөгжүүлэлтийн талаас харсан асуудлын талбар |

Объект хандлагат зохиомжын үндсэн хийгдэх ажил:

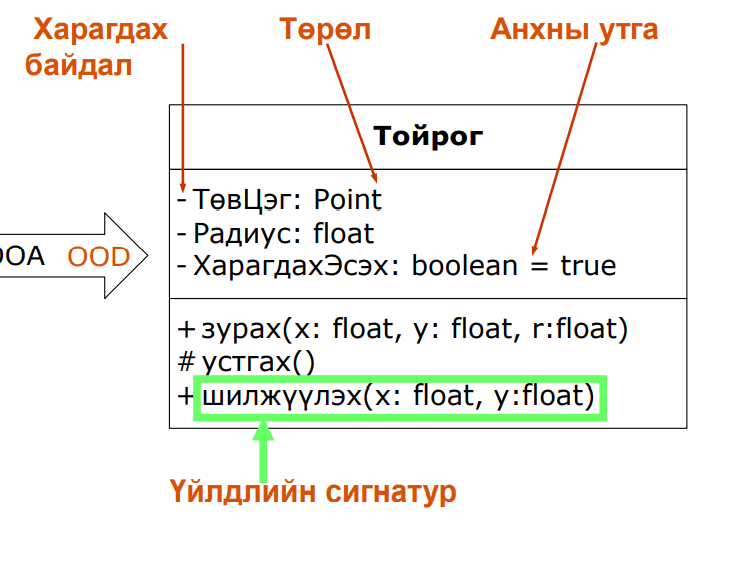
1. Шинжилгээний үед хийгдсэн ажлыг улам өргөтгөдөг.
2. Загварыг нарийвчлан сайжруулах
3. Архитектур гаргаж, бүтэцлэлт хийдэг.

Объект хандлагат загвар нь Шинжилгээнээс зохиомжид тасралтгүй шилжилт. Бүтцийн эвдрэлгүй!

Бид шинжилгээний үед классын диаграммыг дүрслэхдээ:



Объект хандлагат зохиомжын үед нэмэлтээр харагдах байдал, төрөл, анхны утга, үйлдлийн сигнатурыг тусгаж өгч байдаг.



Тус бүрд дэлгэрэнгүй авч үзье

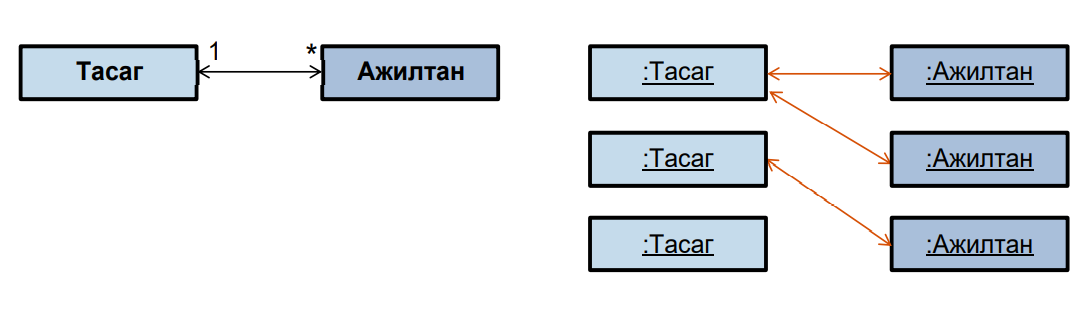
|  |  |  |
| --- | --- | --- |
|  | Шинжилгээ | Зохиомж |
| Харагдах байдал | : Бүх шинжүүд классаас гадагш нууцлагдсан бөгөөд зөвхөн үйлдлээр л уншиж, өөрчилж болно. | • + public: Бусад бүх класст харагдана  • # protected: Класс дотор мөн дэд класст харагдана  • - private: Зөвхөн класс дотор харагдана |
| Үйлдлийн сигнатур | Өгөгдлийн битүүмжлэл болон нууцлалтай байдаг. | • Зохиомж болон хэрэгжүүлэлтийн үед харагдацыг олгосноор битүүмжлэл болон Далдлалтыг салгадаг |
| Холбоосын жолоодлого | Ихэнх холбоос 2 чиглэлтэй байдаг. | Холбоос нэг юм уу аль эсвэл 2 чиглэлтэй байдаг. |

Холбоосын заалтан хувьсагч тусламжтайгаар хэрэгжүүлэх

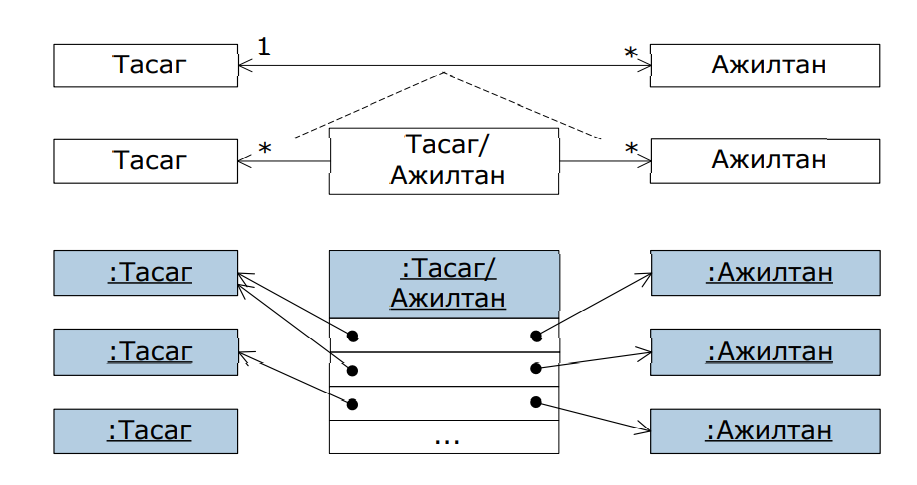
Холбоосын чиглэл болгоны объектийн заалтан хувьсагч ашиглан хэрэгжүүлэхийг хэлнэ.

Харьцааны тоо 0..1 Нэг ширхэг заагч

Харьцааны тоо 1-ээс их бол Олон ширхэг заагчтай гэсэн үг.



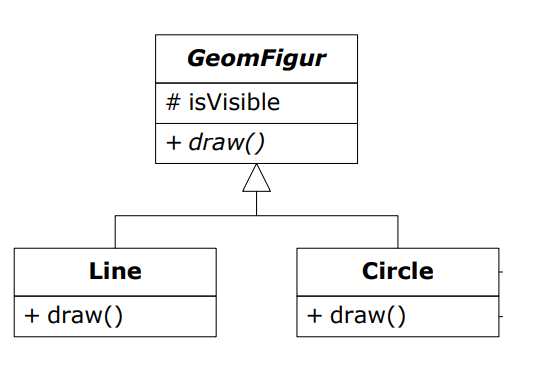
Холбоосыг классын тусламжтайгаар хэрэгжүүлэх

Хоорондох холбоосыг классаар хэрэгжүүлэхийг хэлнэ.

Хийсвэр класс:

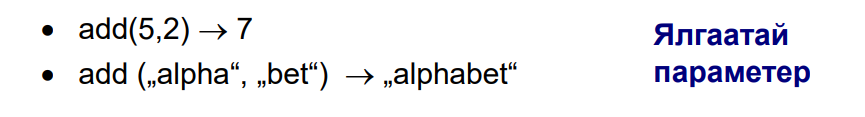
Хийсвэр функцыг агуулдаг объект үүсгэж болдоггүй зөвхөн удамшуулж ашигладаг класс юм. Олон хийсвэр үйлдэлтэй түүнийг дэд классуудад хэрэгжүүлдэг.

Объект хандлагат ЗНХ:



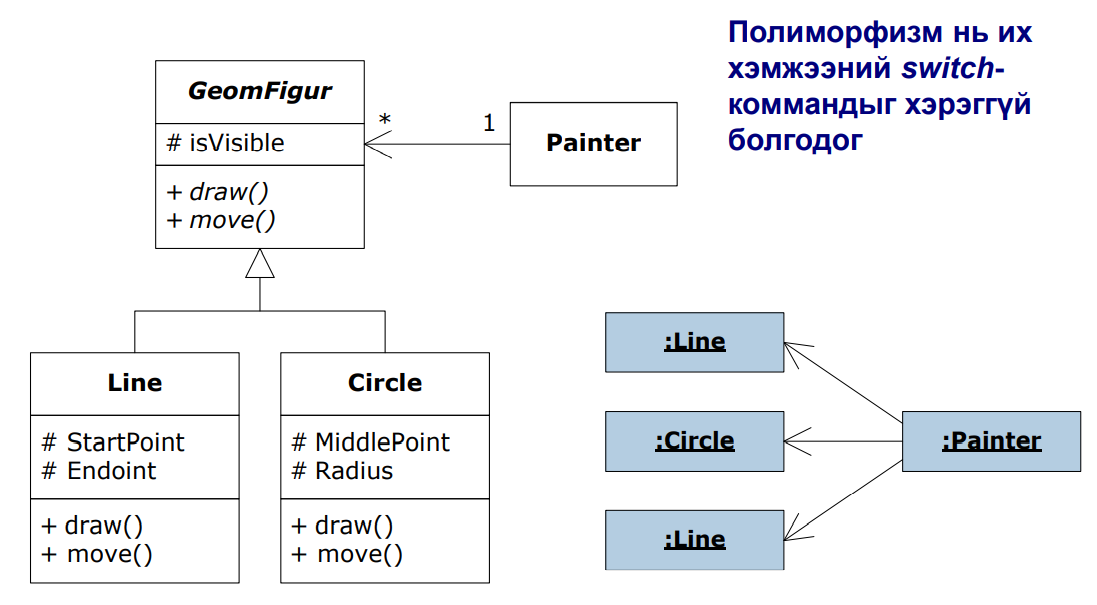
Үйлдлийг дахин тодорхойлох:

Нэг ижил нэртэй харин бичдэсээрээ ялгаатай утга хэрэгжүүлэлт юм.



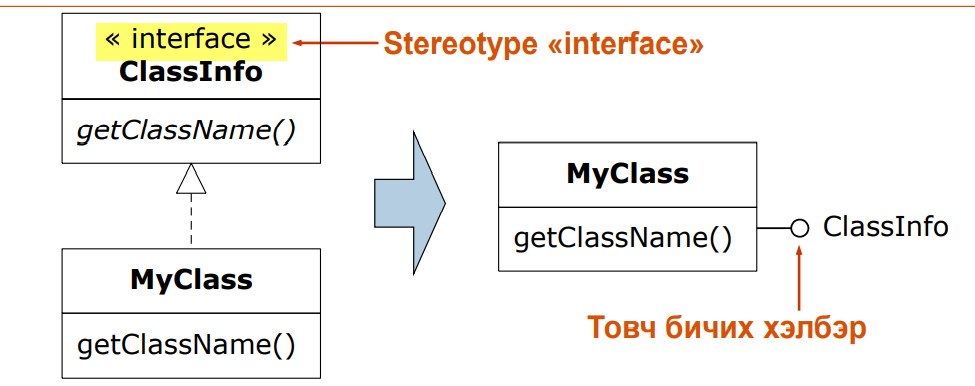
Полиморфизм

Олон төрлийн классын объектуудын хооронд нэг ижил мэдээг явуулахад объект болгон нь өөрийн онцлог, хэв маягаар хүлээх авахыг хэлнэ.



Интерфэйс:

Интерфэйс бол класс хоорондын тохиролцоо юм.

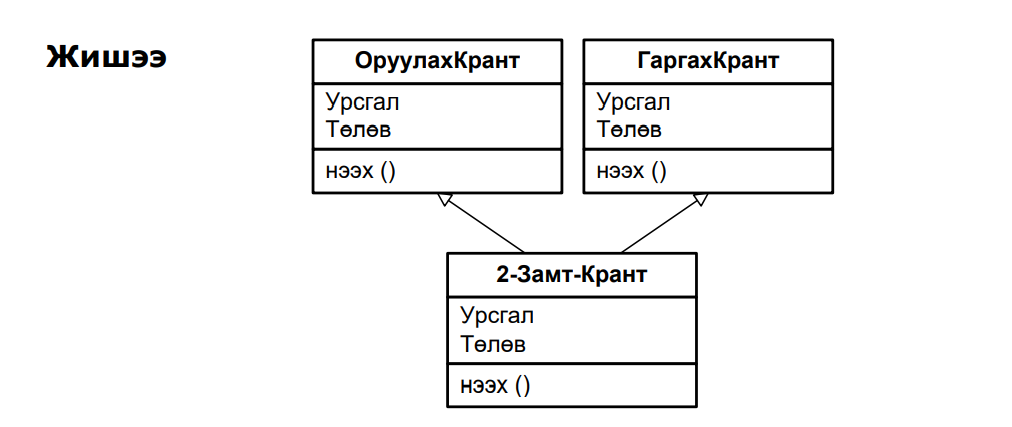


Энгийн болон нийлмэл удамшил

Энгийн удамшил бол мод бүтэц юм. Өөрөөр хэлбэл класс бүр ихдээ нэг эцэг класстай байна.

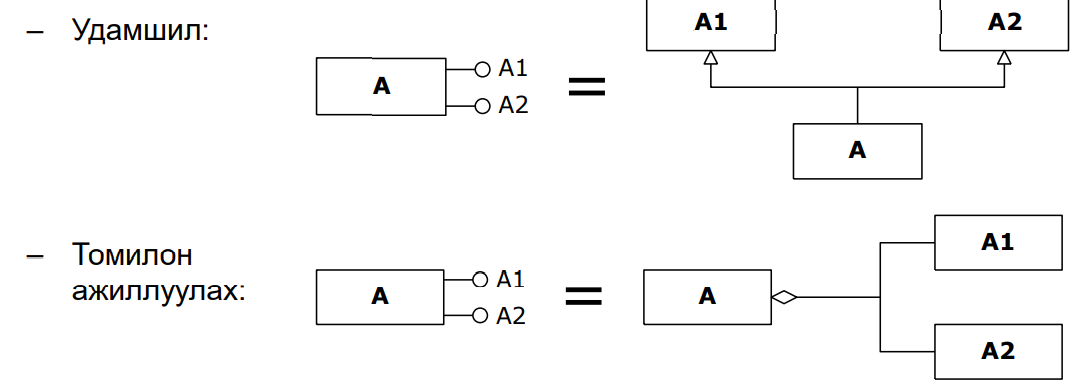
Нийлмэл удамшил:

Класс бүр нэгээс олон эцэг класстай байхыг хэлнэ. Сүлжээ бүтэцтэй байна.



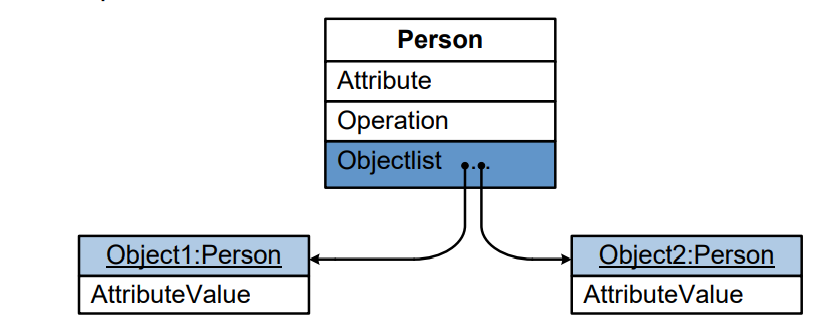
Томилон ажиллуулах нийлмэл удамшлын нэг хувилбар

Объект зурвасыг бие даан ганцаараа бүрэн хэмжээгээр тайлж чадахгүй, харин өөр нэгэн объект дамжуулах механизм юм. Бүрдлийн тусламжтайгаар тэрхүү салгасан классыг холбох

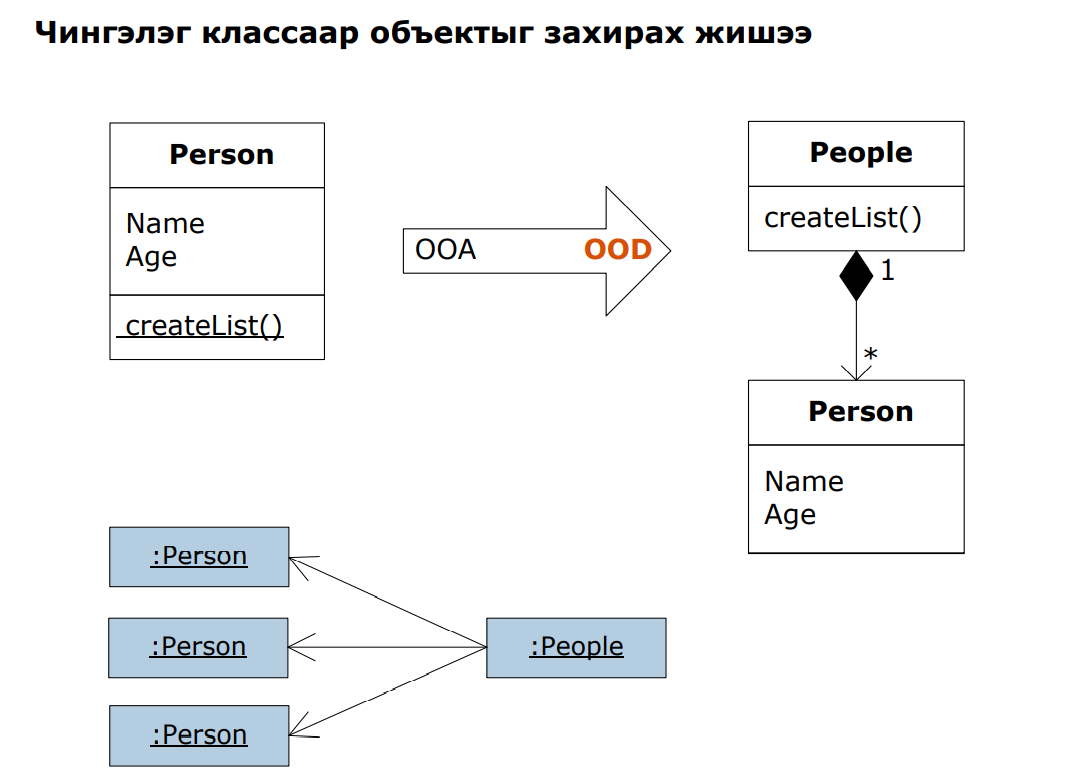


Объект захиран зохицуулалт:

Объект бүр ямар классаас үүссэн гэдгээ мэдэж байдаг, харин класс нь өөрийн объектоо мэддэггүй үүнийг програм дээр статик хувьсагчаар объектоо тоолж чадна. Объект захиран зохицуулалт гэдэг нь класс өөрийн объектыг үүсгэх болон устгахдаа таньж мэднэ гэсэн үг юм. Програм нь статик хувьсагчаар илэрхийлж болно.



Чингэлэг классын тусламжтайгаар объектыг захирах

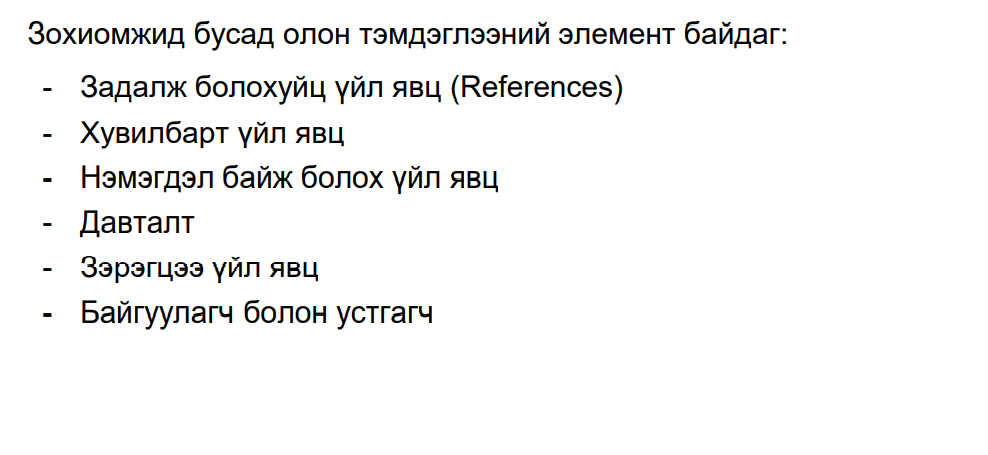


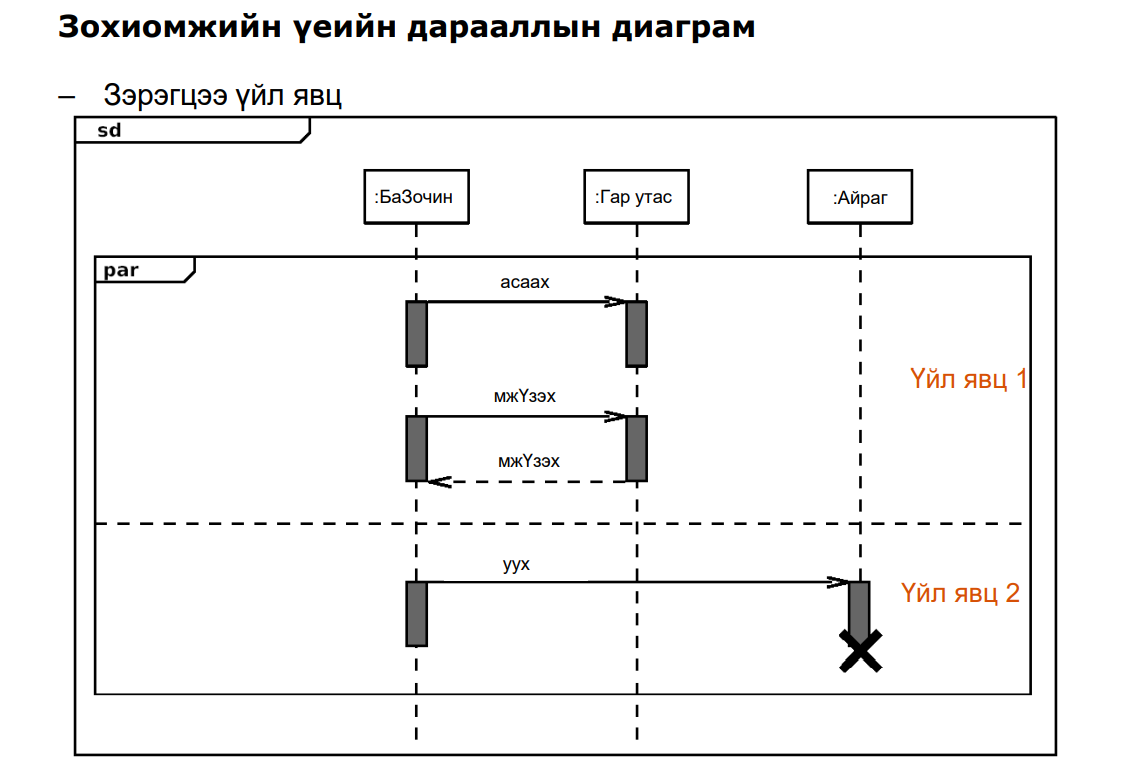
Объектын харилцан ажиллагаа:

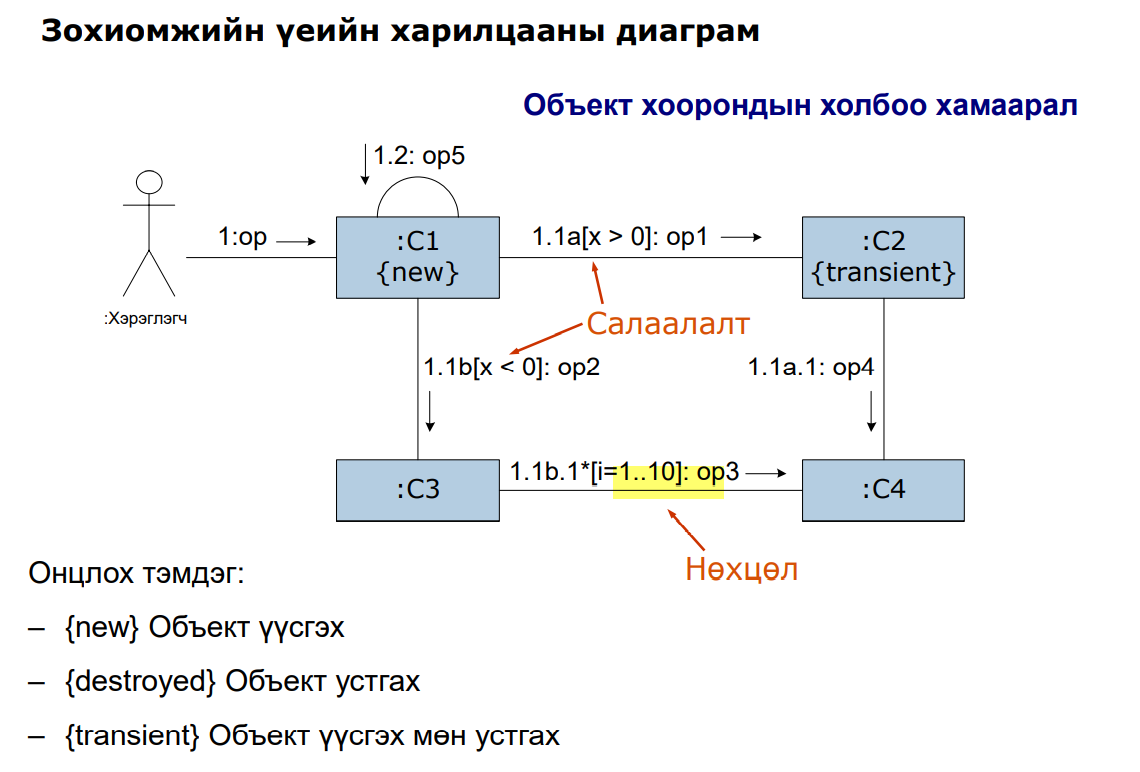
Дарааллын диаграм(Хугацааны талаас илүү харсан) болон харилцааны диаграммаас(Объект хоорондын харьцаа)тогтоно.

Зохиомжын үеийн дарааллын диаграм:

Шинжилгээний үеийн диаграммыг өргөтгөж өгдөг.







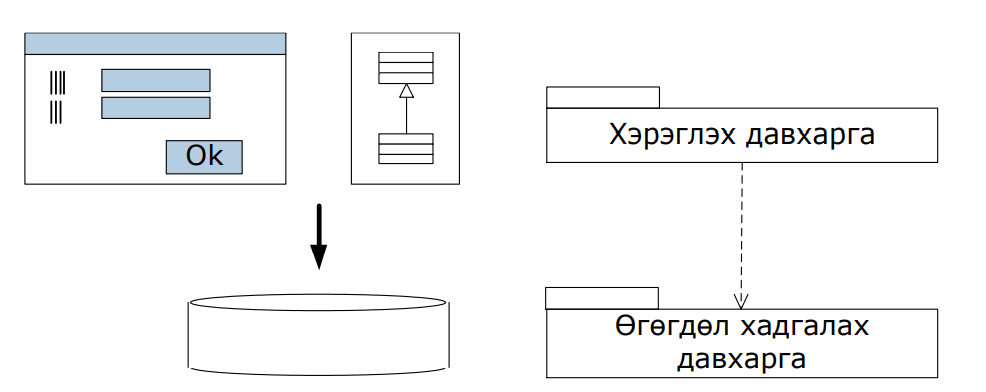
Архитектурын зохиомж:

Програм хангамжын архитектурын зорилго бол бизнес загварт тодорхой хязгаар нөхцөлийг авч үзэн зохиомжийн загварыг хөгжүүлэх юм.

Програм хангамж урт удаан оршин байхын тулд сайн архитектуртай байх явдал юм. Систем аль болох хоорондоо бага хамааралтай бүтэцлэх шаардлагатай байх ёстой. Архитектур нь хоёр давхаргат, гурван давхаргат, олон давхаргат архитектур байдаг.

Хоёр давхаргат архитектур нь

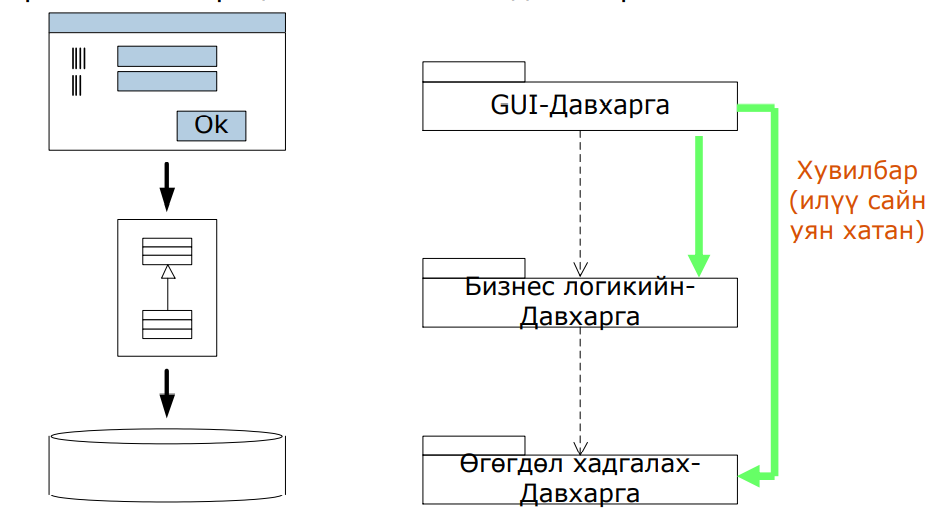
Зөвхөн хэрэглэгчийн харьцах хэсэг болон бизнес загвараас(Хэрэглэгчийн загвар) болон өгөгдлүүдийг хадгалах өгөгдлийн загвараас бүрддэг. Жишээ нь

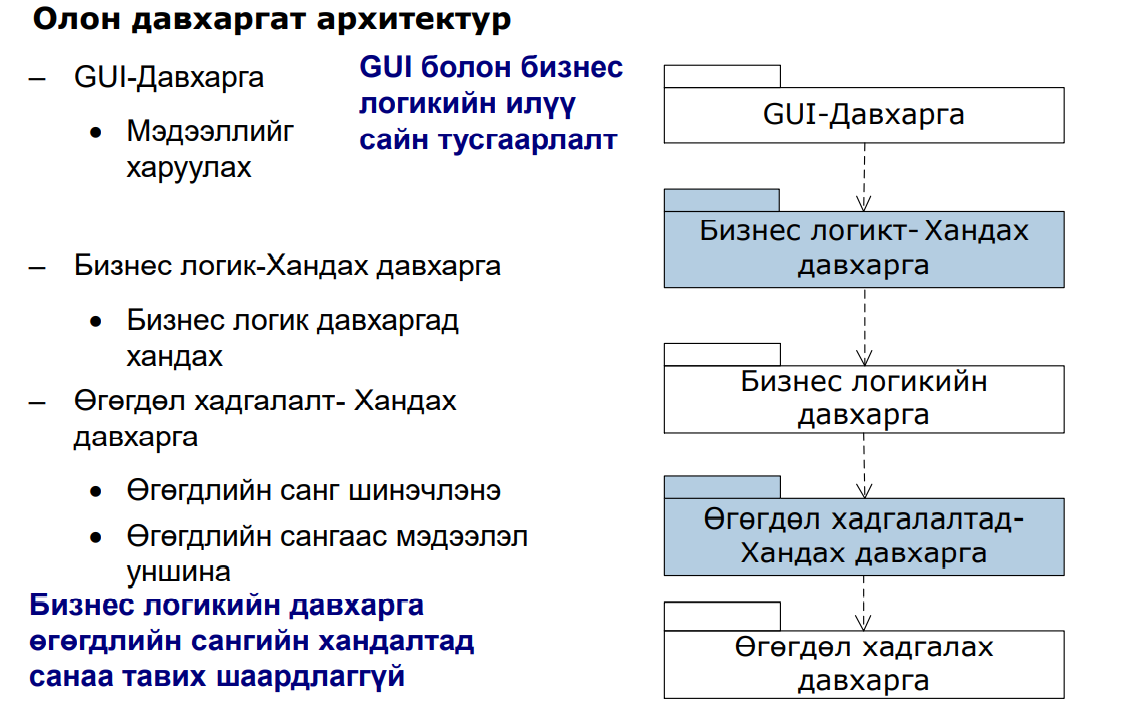


Аливаа зүйлд сул тал байдаг бизнесийн ойлголт хэрэглэх интерфэйё цугтаа байдаг. Системийг бага хамааралтай болтол жижиглэх ёстой байдаг.

Гурван давхаргат архитектур бол:

Бизнес ойлголт болон хэрэглэгчийн интерфэйсийг салгаж өгсөн. Хэрэглэгчийн харьцах хэсгийг өөрчлөхөд хялбар байдаг.



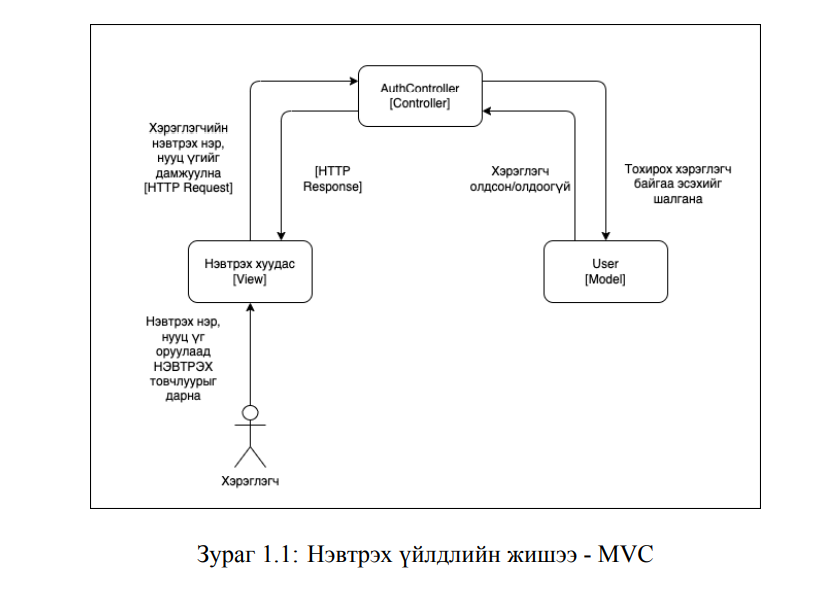


Нэмэлт мэдээлэл:

* 1. Програм хангамжийн архитектур Архитектур нь системийн зураг төсөл (blueprint)-г илэрхийлдэг. Энэ нь системийн нарийн төвөгтэй байдлыг зохицуулж, бүрэлдэхүүн хэсэг хоорондын харилцаа болон зохицуулалтын механизмыг зохион байгуулдаг. • Энэ нь гүйцэтгэл, аюулгүй байдал зэрэг чанарын нийтлэг шинжүүдийг оновчтой болгох замаар техникийн болон үйл ажиллагааны шаардлагуудад тусгагдсан бүтэцлэгдсэн шийдлийг тодорхойлдог. • Цаашилбал, энэ нь програм хангамжийн хөгжүүлэлттэй холбоотой байгууллагын талаарх чухал шийдвэрүүдийг багтаадаг бөгөөд эдгээр шийдвэр тус бүр нь чанар, тогтвортой байдал, гүйцэтгэл, эцсийн бүтээгдэхүүний нийт амжилтанд ихээхэн нөлөөлдөг.
  2. 1.2 MVC (Model-View-Controller) архитектур MVC архитектур нь хэрэглэгчийн интерфэйс, өгөгдөл, бизнес логикыг хэрэгжүүлсэн өргөн хэрэглэгддэг програм хангамжийн зохиомжийн хэв загвар юм. (software design pattern) Энэ нь програмын харагдах хэсэг болон бизнес логикыг тусад нь байлгадаг.
  3. 3 1. Model: Өгөгдлийн сангаас унших, бичих

2. View: Хэрэглэгчийн харьцах хэсэг

3. Controller: Хэрэглэгчийн харьцах хэсгийн хүсэлтийг боловсруулах, хамаарах үйлдлийг гүйцэтгэх, model-тэй харьцах Хэрэглэгч тодорхой үйлдлийг хийхэд жишээ нь “Нэвтрэх” товчлуурыг дарах үед viewс хамаарах controller-г дуудна. Controller хэрэглэгчийн нэвтрэх нэр, нууц үгтэй тохирох хэрэглэгчийн бичилт байгаа эсэхийг шалгах User model-н үйлдлийг дуудна. Хүсэлтийн үр дүнд view өөрчлөгдөнө (re-render).

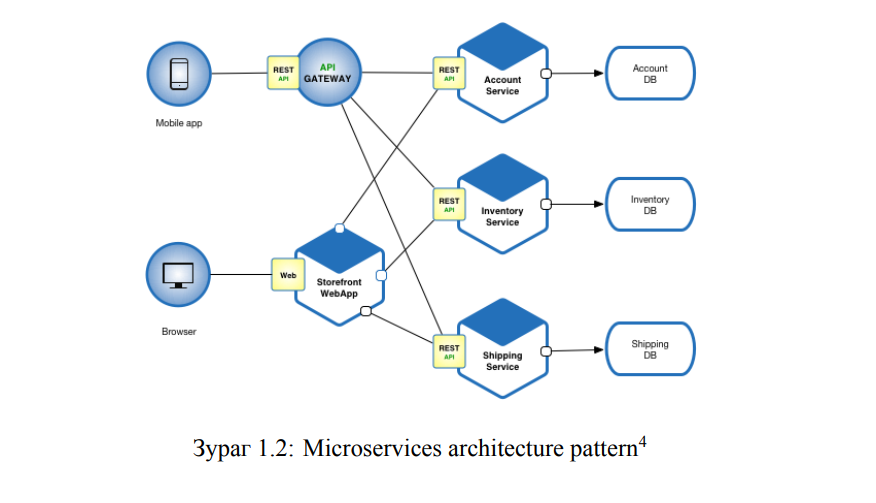


Давуу тал • Хөгжүүлэлтийн процесс хурдан. Гурван түвшинд хувааснаар хөгжүүлэгчид эдгээр хэсгүүдийг хувааж хөгжүүлэх боломжтой.

4 • Multiple views. Аливаа model-д зориулан хэд хэдэн view компонентуудыг зохион байгуулах боломжтой. • Asynchronous дэмждэг. MVC архитектур нь Javascript болон түүний framework-уудтай маш сайн зохицдог бөгөөд энэ нь хөгжүүлэгчдэд илүү хурдан ачаалах вебийг бүтээх боломжийг олгодог. • Өөрчлөхөд хялбар. Хэрэглэгчийн харьцах хэсэг болон бизнес логикийг салгаснаар аль нэг давхаргад шууд өөрчлөлт оруулах боломжтой. • MVC model нь формат хийгээгүй өгөгдлийг буцаадаг. Формат хийгээгүй өгөгдлийг буцааснаар MVC нь танд өөрийн view engine-г бий болгох боломжийг олгоно. Жишээлбэл, HTML ашиглан ямар ч төрлийн өгөгдлийг форматлах боломжтой боловч нь MVC-н тусламжтайгаар Macromedia Flash эсвэл Dream viewer ашиглан өгөгдлийг форматлах боломжтой. Ижил бүрэлдэхүүн хэсгүүдийг ямар ч интерфэйстэй дахин ашиглах боломжтой болгосноор хөгжүүлэгчдэд тустай. • SEO friendly. MVC платформ нь SEO-д ээлтэй веб програмуудыг хөгжүүлэхэд ихээхэн дэмжлэг үзүүлдэг. Тодорхой програмаас илүү олон хандалтыг бий болгохын тулд MVC нь SEO-д ээлтэй RESTful URL-уудыг хөгжүүлэх хялбар аргыг санал болгодог. • TTD (Test-Driven development)-г дэмждэг. MVC-ийн гол давуу тал нь туршилтын үйл явцыг ихээхэн хөнгөвчлөх явдал юм. Том хэмжээний програмыг олон түвшинд хувааж дибаг хийхэд хялбар болгодог ба ингэснээр програмыг нэгжийн тестийн хамтаар хөгжүүлэх боломжийг олгоно.1 1.2.2 Сул тал • Нарийн төвөгтэй байдал (Complexity). Цаг хугацаа өнгөрөхөд MVC архитектурын загвар нь шууд бусаар шинэчлэгддэг бөгөөд энэ нь шийдлийг улам хүндрүүлдэг. Нэмж дурдахад 1Benefits of using MVC, https://www.geeksforgeeks.org/benefit-of-using-mvc

5 хэрэглэгчийн интерфэйсийн код ихсэнээр дибаг хийхэд улам төвөгтэй болдог. • Ойлгоход төвөгтэй. MVC архитектур нь үнэн хэрэгтээ ойлгож суралцахад тийм ч хялбар биш. Нарийн төвөгтэй бүтэц болон давтамжтай шинэчлэлт нь системийн бүхий л бүрэлдэхүүн хэсэг болон хэрэглэгчийн харьцах хэсгийн код дээр тохиодог. • Ойр ойрхон өөрчилж, шинэчлэхэд зардал өндөр. Хэрэв model нь давтамжтайгаар өөрчлөгдөөд байвал view-г ч өөрчлөх хүсэлтүүд ихээр гарч болох юм. Цаашилбал view нь хэрэглэгчийн харьцах хэсэг тул өөрчлөхөд хугацаа их шаардагдаж, үр дүнд view компонентуудад шинэчлэл бүр дээр алдаа гарах магадлал бий болно. • Үйлдлүүд (methods) нь хориглолттой байх ёстой. MVC архитектурын 3 дахь бүрдэл хэсэг нь Controller. View компонентуудад тодорхой үзэгдэл тохиоход үйлдлүүдийг дуудаж ажиллуулдаг. Гэхдээ эдгээр нь хандалтын ямар ч хориглолтгүй байдаг.

1.2.3 Дүгнэлт MVC архитектурын давуу тал нь сул талаа нөхдөг. Сурах, ойлгоход хэцүү ч боломжгүй зүйл биш. Мөн нөгөөтэйгүүр MVC архитектурт суурилсан програмуудаар үйлчлүүлэгчдийнхээ хэрэгцээг хангадаг олон веб хөгжүүлэлтийн компаниуд байдаг. 1.3 Microservices архитектур Microservices архитектур нь том хэмжээний нарийн төвөгтэй програмыг хурдан, алдаагүй, илүү үр дүнтэй байдлаар хөгжүүлэхэд зориулагдсан бөгөөд системийн үйл ажиллагааг хэд хэдэн бие даасан сервисүүдэд хувааж зохиомжилсон системийн архитектур юм. Сервис бүр нь өөрийн өгөгдлийн сантай байх бөгөөд өөр хоорондоо энгийн API-н тусламжтай холбогдон ажилладаг. 3



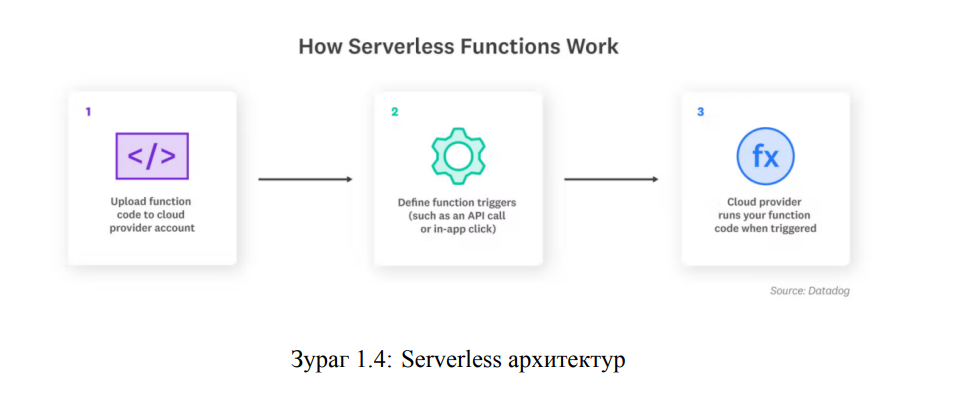
MVC архитектур дээр ашигласан хэрэглэгч нэвтрэх жишээн дээр харцгаая. Хэрэглэгч нь нэвтрэх үйлдэл нь AuthService-т хамаарах бөгөөд өөрийн AuthDB гэсэн өгөгдлийн сантай байна. Үүнд хэрэглэгчийн мэдээлэл хадгалагдана. Үүнээс гадна MedicalService гэсэн микросервисийг дүрсэлсэн байна.

Давуу тал • Бүтээмжийг нэмэгдүүлдэг (Improved productivity). Системийг жижиг хэсгүүдэд хуваан хөгжүүлэлт хийж, бие даан ажиллуулах боломжтой байдаг. Хэсэг бүрийг ялгаатай хэл, технологи ашиглан хэрэгжүүлж болно. • Тэсвэрлэх чадвар өндөр (Better resiliency). Microservice дээр тулгуурласан архитектур нь ажиллагааны явцад гарах асуудлыг илрүүлэх, түүнийг шийдвэрлэх процессыг илүү хялбар болгосон. Алдаа гарсан жижиг хэсгийг олж илрүүлж засаад зөвхөн тухайн дэд сервисийг дахин ажиллуулахад /redeploy/ хангалттай. • Өргөтгөх боломж (Better scalability). Жижиг хэсгүүд дээрх өөрчлөлт, сайжруулалтыг мөн адил бие даасан байдалтайгаар хөгжүүлдэг. • Бизнесийн үйл ажиллагааг оновчтой болгодог (Optimize business functionality). Бүхэл систем биш дэд хэсэг дээр илүү анхаарч, бүрэлдэхүүн хэсгүүдийн хэрэгцээг хангаснаар бизнесийн үйл ажиллагааг илүү оновчтой болгодог.5

Сул тал • Хөгжүүлэлтийн багт Monolithic програмуудаас илүү олон төрлийн нарийн төвөгтэй байдлыг бий болгодог. – Нэгт. Сервис хоорондын харилцаа нь төвөгтэй байж болно. Програм нь хэдэн арав эсвэл хэдэн зуун ялгаатай сервисүүдийг агуулдаг бөгөөд тэдгээр нь бүгд хоорондоо аюулгүй холбогдож ажиллах ёстой байдаг. – Хоёрт. Дибаг хийх нь илүү төвөгтэй. Програм нь хэд хэдэн microservice-үүдийг агуулах бөгөөд тэдгээр нь тус бүрдээ log-уудтай, асуудлыг олж илрүүлэх нь төвөгтэй байж болох юм.

Гуравт. Unit тестийн хувьд хялбар байж болох ч integration тестийн хувьд тийм биш юм. Компонентууд нь тархсан байдалтай байдгаас хөгжүүлэгчид өөрсдийн төхөөрөмж дээр системийг бүхэлд нь тестэлж чаддаггүй. • Интерфэйсийн хяналт маш чухал. – Microservice бүр нь өөрийн API-тай бөгөөд програм нь эдгээр дээр тулгуурлан тогтвортой ажилладаг. Microservice-тэй харьцаж буй гадаад системд нөлөөлөхгүйгээр хялбархан өөрчлөлт хийх боломжтой ч API (интерфэйс) дээрх өөрчлөлт нь түүнтэй холбогдож байгаа бусад системүүдэд нөлөө үзүүлвэл энэ нь асуудлыг бий болгоно. – Microservices архитектурын загвар нь системийн үйл ажиллагаанд чухал ач холбогдолтой олон тооны API-г бий болгодог тул интерфэйсийн хяналт нь чухал юм. • Урьдчилсан зардал (up-front cost) өндөр байж болно. – Танай байгууллагад Microservices архитектурыг ажиллуулахын тулд танд аюулгүй байдал, засвар үйлчилгээний дэмжлэг бүхий хангалттай байршуулах дэд бүтцээс гадна бүх үйлчилгээг ойлгож, удирддаг чадварлаг хөгжүүлэлтийн багууд хэрэгтэй. – Хэрэв танд эдгээр зүйлс байгаа бол Microservice-т шилжихэд гарах зардал бага байж болох ч одоогоор Monolithic архитектурыг ажиллуулж байгаа ихэнх аж ахуйн нэгжүүд шилжихийн тулд шинэ дэд бүтэц болон хөгжүүлэгчийн нөөцөд хөрөнгө оруулалт хийх шаардлагатай болно.

6 1.3.3 Дүгнэлт Microservice архитектурын давуу болон сул талууд нь уламжлалт Monolithic архитектураас эрс ялгаатай бөгөөд энэ загвар нь бүх байгууллагад тохиромжтой биш юм. Гэсэн хэдий ч тодорхой шалтгаануудын улмаас энэхүү модульчлагдсан архитектурт шилжиж байна. Энэ 6The Disadvantages of Microservices, https://solace.com/blog/microservices-advantages-and-disadvantages 9 нь илүү олон байгууллагууд илүү хурдан, хялбар, уян хатан програмын хэрэгцээг бий болгож байгаа явдал юм. 1.4 Serverless архитектур Serverless архитектур нь програм хангамжийн дэд бүтцийг (infrastructure) удирдан зохицуулахгүйгээр хөгжүүлэлтийг гүйцэтгэж, системийг ажиллуулах арга. Систем нь сервер дээр ажиллах хэдий ч серверийн бүхий л менежментийг AWS, Firebase, Azure зэрэг cloud platformууд хариуцдаг



Давуу тал • Хямд. Зөвхөн өөрт нийцэх cloud үйлчилгээг худалдан авч ашиглахад л хангалттай. • Уян хатан байдал. Serverless загвар нь таны оролцоогүйгээр өргөжих боломжтой. • Нарийвчлал. Хөгжүүлэгчид үүнийг дэмждэг back-end архитектураас илүүтэй тодорхой функц дээр анхаарлаа төвлөрүүлэх боломжтой. Энэ нь илүү цэвэрхэн кодыг бий болгодог. • Хурд. Хөгжүүлэгчид серверийн багтаамжийг тооцоолох, хуваарилах дээр санаа зовж хугацаа алдахгүй.9 1.4.2 Сул тал • Аюулгүй байдал. Өөрийн өгөгдөл мэдээллийг бусад компанид хадгалуулах явдал. Serverless шийдлийг ашигладаггүй байгууллагуудын 60 гаруй хувь нь энэ загварын аюулгүй байдлаас болдог. • Нууцлал. Таны мэдээлэл бусад хүмүүс ашиглаж болохуйц cloud орчинд байршдаг. • Нарийн төвөгтэй байдал. Хэрэв ямар нэг зүйл зохих ёсоор ажиллахгүй байвал асуудлыг олж тогтооход хэцүү. • Гэрээ. Зарим борлуулагчид үйлчлүүлэгчтэйгээ урт хугацааны гэрээ байгуулдаг.1

Дүгнэлт

Энэхүү лекцүүдийн тэмдэглэл бичснээр лекцийн ойлголтуудаа бататгаж чадсан. Хамгийн гол чухал ойлголт болох холбоос болон түүний ангилал, удамшил , хийсвэр удамшил гэх тодорхойлтыг ойлгож, статик болон динамик загвар, програм хангамжийн системийн шинжилгээний үе ба зохиомжын үеийг загварчлалын нэгдсэн хэлээр дүрсэлж сурсан. Объект хандлагат хэл нь аливаа бодит зүйлийг хийсвэр түвшинд хийхэд тохиромжтой байдаг. Програм хангамжын бүтээгдэхүүнийг боловсруулах үед бүтцийн эвдрэл гардаггүй. Сайн програм хангамжийн архитектур нь урт хугацааны програмын бүтээгдэхүүн болоход оршино.