Зохиомжийн загварууд 3

Програм хангамжийн архитектур (Software Architecture)

2012

С. Бадрал

Агуулга

- Нэр томьёо
- Facade pattern
- Observer pattern
- MVC pattern
- Дүгнэлт

Нэр томьёо

- Pattern = Загвар / паттерн
- Facade = Фасад
- Observer = Ажиглагч
- MVC = Model View Controller
- Event = Үзэгдэл
- Couple = Хэрээс
- Concret = Конкрет (тодорхой)

Facade pattern

Зорилго

- Төвөгтэй дэд системийн зурвасууд эсвэл харилцан хамааралтай объектуудын олонлог руу хандах хандалтыг хялбарчилна. Нэгдсэн нэг зурвас нийлүүлнэ.

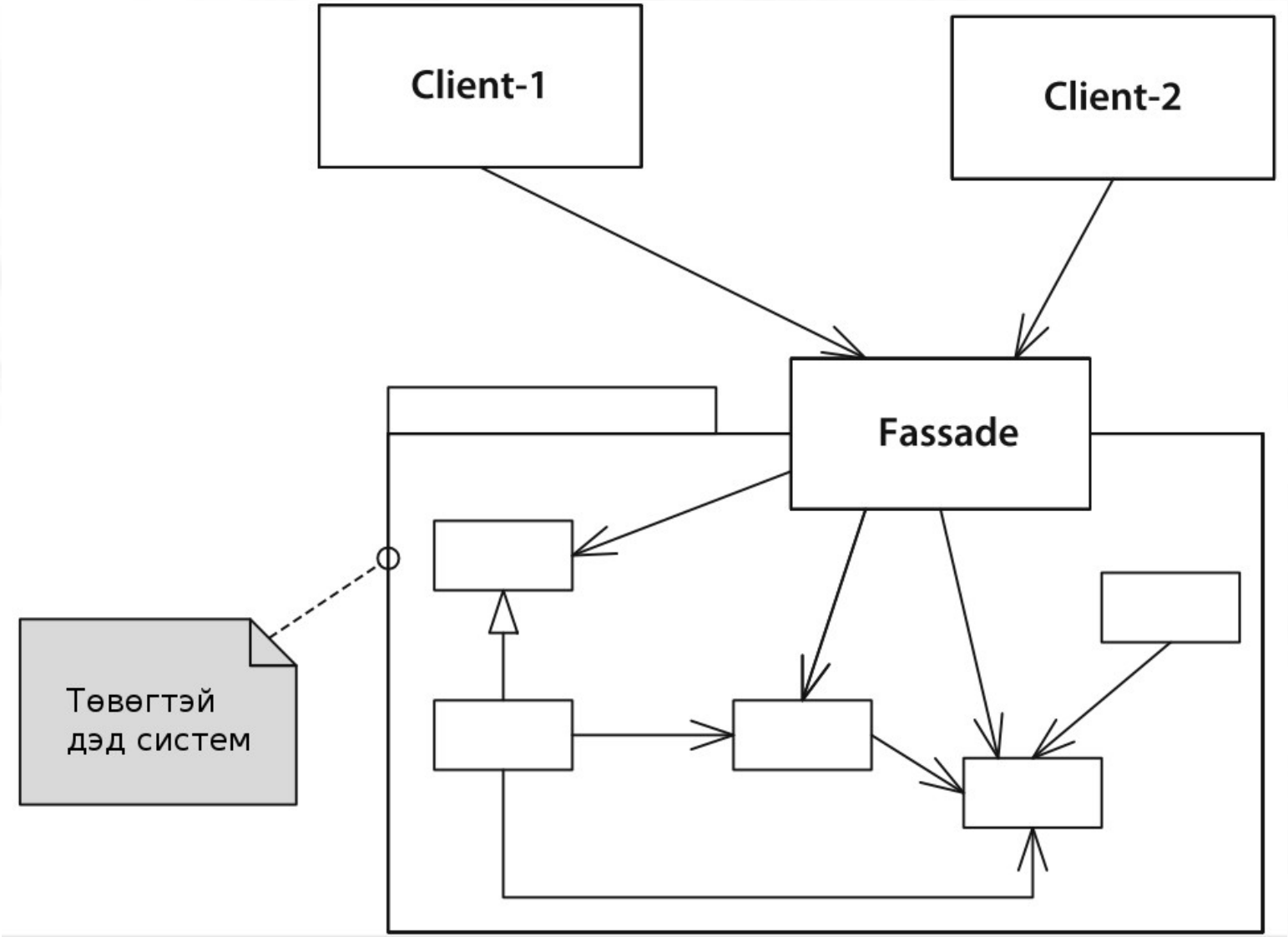
Асуудал

- Та дотоод бүтэц нь маш төвөгтэй дэд систем уруу хандах шаардлагатай болсон гэж бодъё. Түүний тулд та тэрхүү дэд системийн дотоод бүтцийг аль болох нарийвчлан мэдэхийг хүсэхгүй.

Хэзээ хэрэглэх вэ?

- Төвөгтэй дэд систем рүү хандах шаардлагатай бол
- Клиент болон хийсвэрлэлийн хэрэгжүүлэгч класс хооронд олон хамаарал байвал хэрээсийг (couple) багасгахаар
- Төвшинт загвар ашиглаж байвал хандалтын цэгүүдээ зааглахад

Facade pattern



Observer pattern

Зорилго

- Observer pattern нь ямар нэг тодорхой объект өөрчлөгдөхөд автоматаар нэг эсвэл хэд хэдэн объектод өөрсдийн төлөвийг нь тааруулахын тулд сонордуулах боломжийг олгоно.

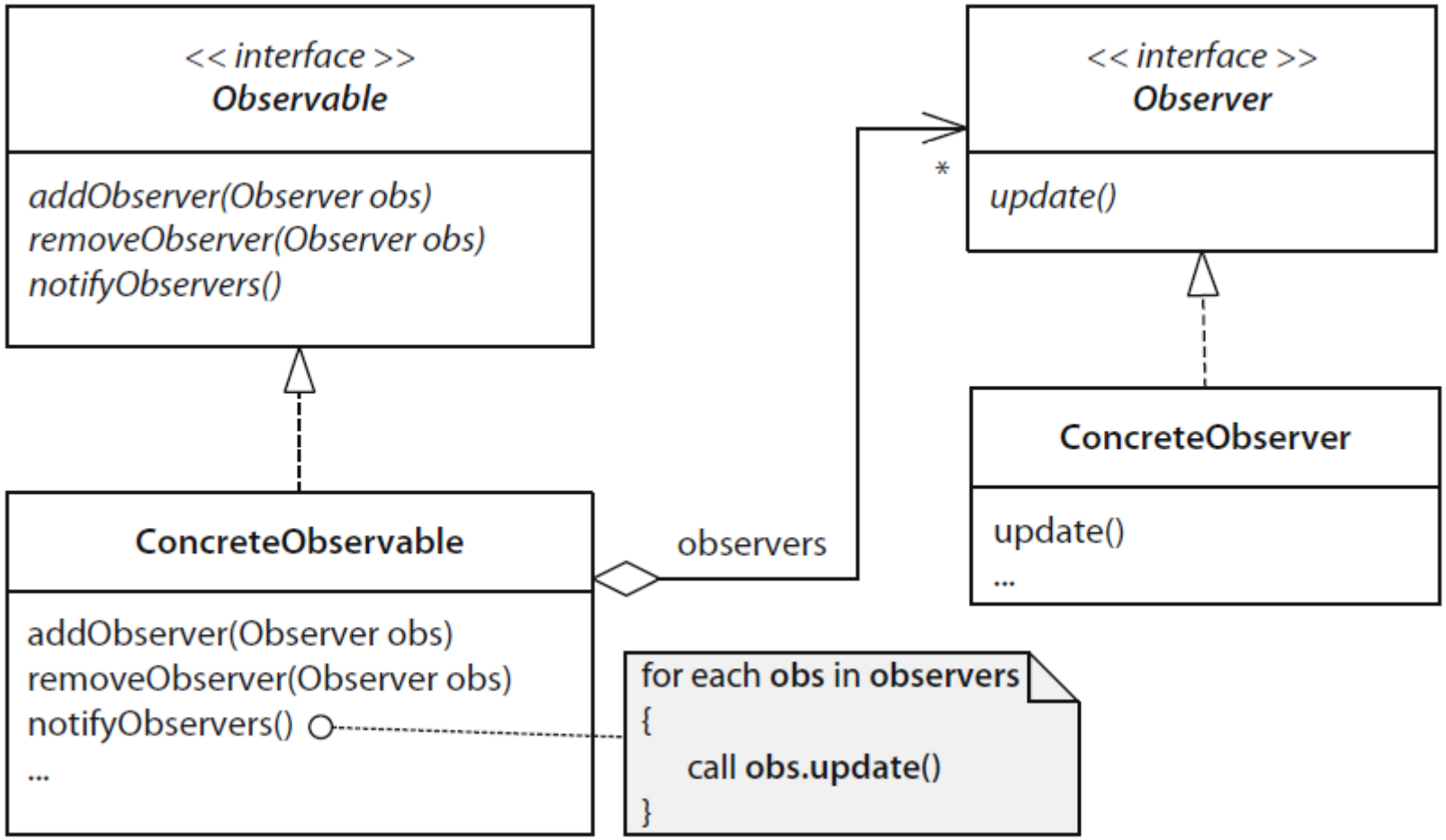
Асуудал

- Ямар нэгэн объектын төлөв өөрчлөгдөнгүүт өөр объект төлөвөө автоматаар тааруулах ёстой.

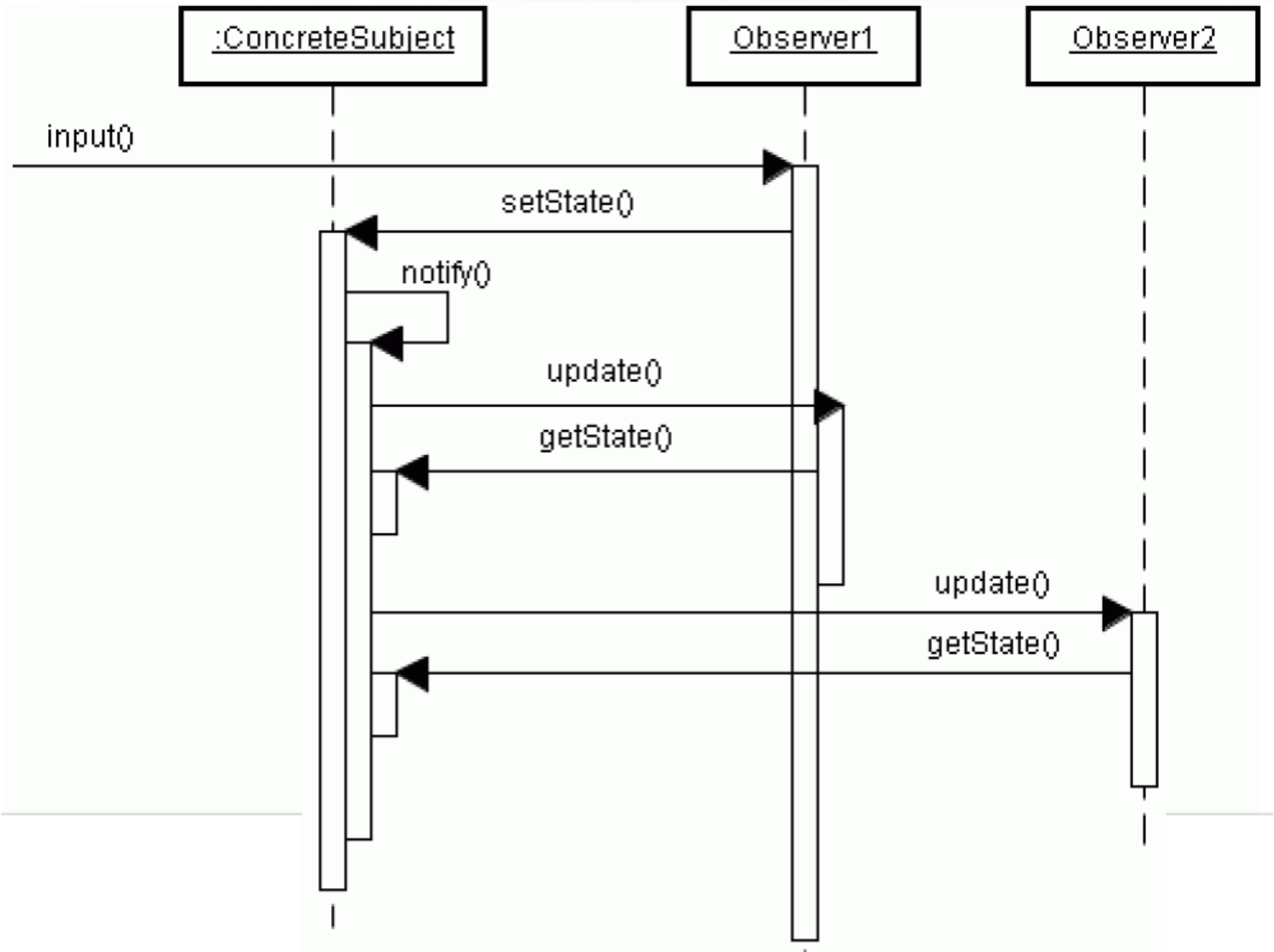
Хэзээ хэрэглэх вэ?

- Хэрэв нэг объект нөгөө объектоос хараат бол
- Хэрэв нэг объектын өөрчлөлтөөр бусад олон объектод өөрчлөлт оруулах шаардлагатай бол
- Хэрэв нэг объектын өөрчлөлтүүдийг бусад объект руу (тэдний тухай ямар нэг мэдээлэлгүйгээр) сонордуулах шаардлагатай бол

Observer pattern



Observer pattern



Observer pattern

- Хэрэв субъектын (ConcreteObservable эсвэл ConcreteSubject) төлөв өөрчлөгдвөл тэр бүх өөрчлөлтүүд нь бүх ажиглагчид (ConcreteObservers) мэдээлэгдэнэ. (notify)
- Ингээд ажиглагч тус бүр субъектын төлвийг асууж түүнтэй өөрийн төлөвийг тааруулна. (update)
- Субъект нь өөрийн ажиглагчийг бүртгэх ба хасах зохицуулагч методуудыг агуулна. (add, remove)
- Конкрет ажиглагч өөрийн ажиглаж буй (конкрет) субъектыг танина.

  Хэрэглээ:

  Данс хооронд гүйлгээ хийх, хүснэгтэн баримтад диаграмм үүсгэх, ... гэх мэт

Интерактив системүүд

Интерактив системүүдийн ерөнхий загварууд нь:

- Model-View-Controller (MVC)
- Presentation-Abstraction-Control (PAC)

Нөхцөл байдал

- Уян хатан хэрэглэгчийн гадаргуутай системүүд
- Хэрэглэгчийн гадаргуу байнга өөрчлөгдөх (Жишээ нь: шинэ үйл ажиллагааны улмаас цэс өргөтгөх)
- Үйл ажиллагааны цөм ба ХГ (GUI) нягт холбосноор өөрчлөлт хийгдэхэд төвөгтэй бөгөөд алдаа их гардаг

Интерактив системүүдийн шаардлага

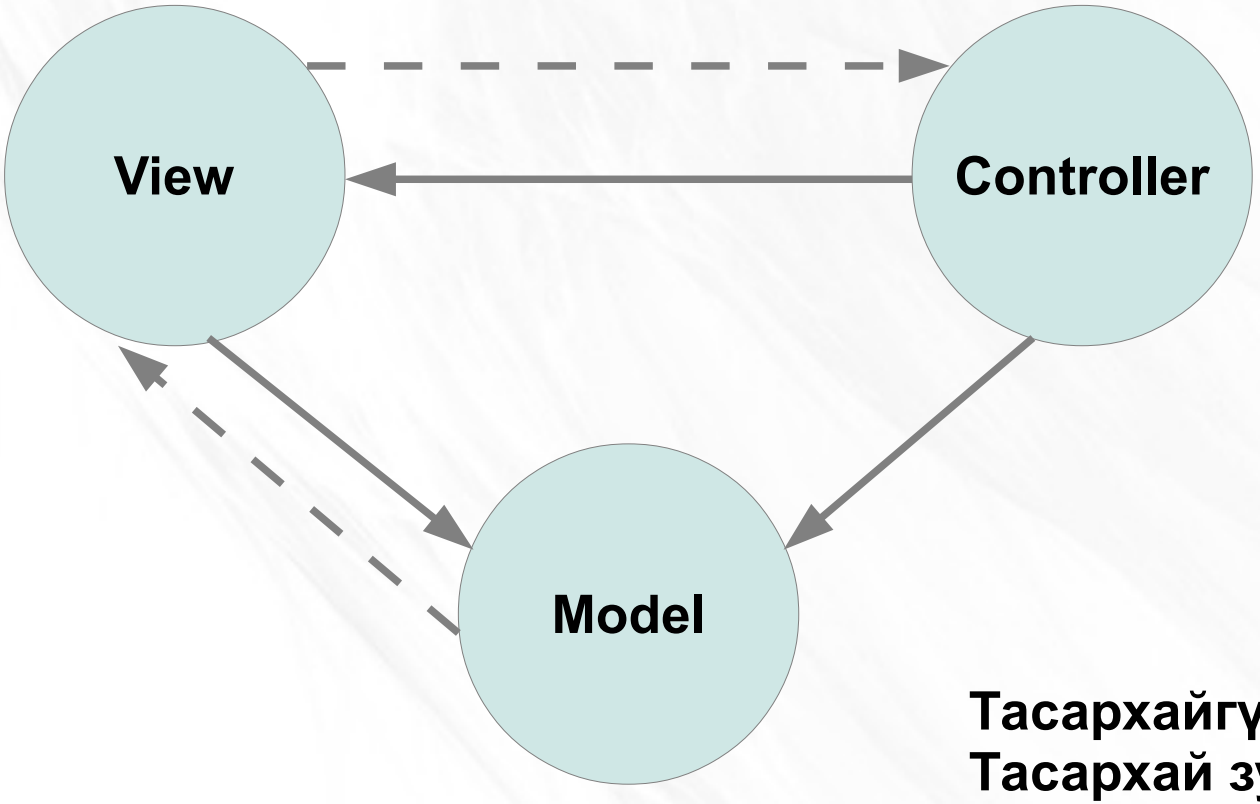
- Ижил өгөгдөл өөр өөр цонхонд өөр өөр техникээр (график, текст) харагдах.
- Өгөгдлийн төлөв өөрчлөгдөнгүүт өгөгдөл харуулалт ба программын харьцаанд тусах
- Хэрэглэгчийн гадаргуун өөрчлөлт хялбар байх
- Системийг өөр “Look and Feel” рүү шилжүүлэх нь функц буюу үйл ажиллагаанд нөлөөлөхгүйгээр хийгдэх боломжтой байх

- Санаа: Хийсвэр загвар (abstract model), GUI харагдац (view), удирдлагыг (control) салгах

Separation of Concerns

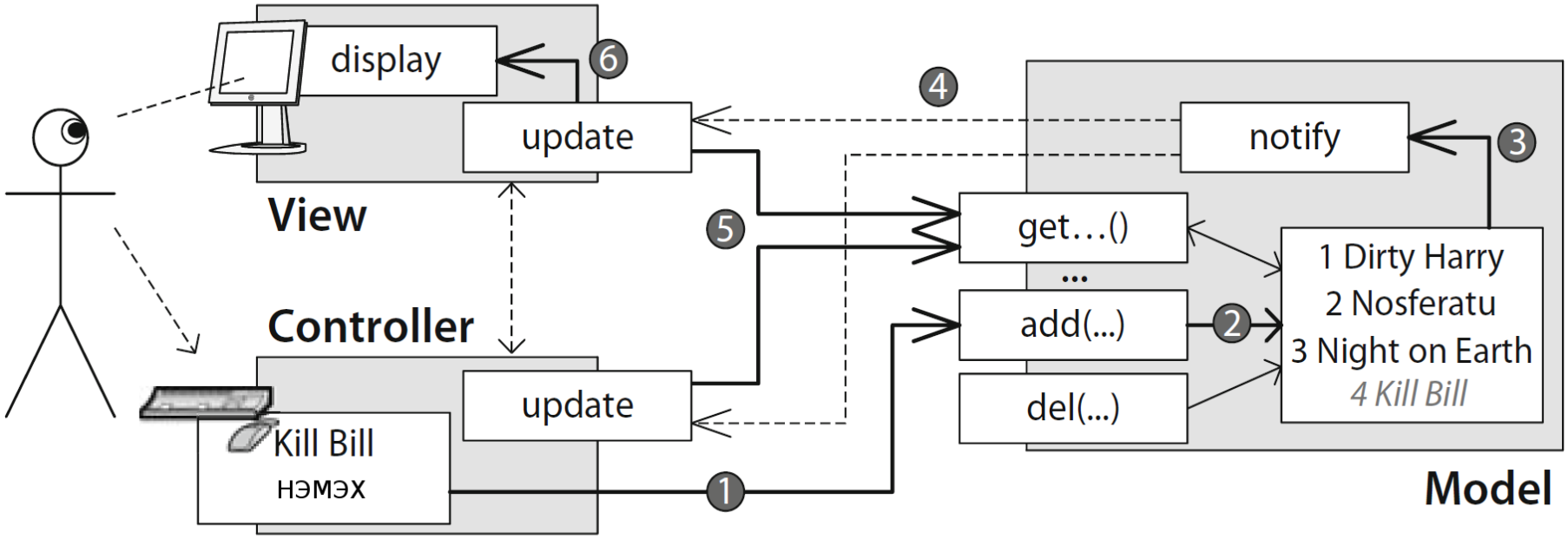
MVC pattern

- Хамгийн өргөн тархсан загвар
- Веб програмчлалд хамгийн түгээмэл хэрэглэж байна



Тасархайгүй зураас: Шууд
Тасархай зураас: Шууд бус

MVC pattern



View - Ажиглагч: (1) Дуудалт (2) Модел өөрчлөлт
(3)+(4) Сонордуулга (5)+(6) Шинэчлэл

Модел (Model)

- Системийн төлөвийг мэднэ
- Мэдээллийг буцаана (ихэнхдээ харагдац рүү)
- Өөрчлөлтөд хариу өгнө (ихэнхдээ Удирдлагаас өдөөгдөнө)
- Үзэгдэлд суурилсан системүүдэд Модел нь “Observable” дүрийг авна (Харагдац нь observer болно)

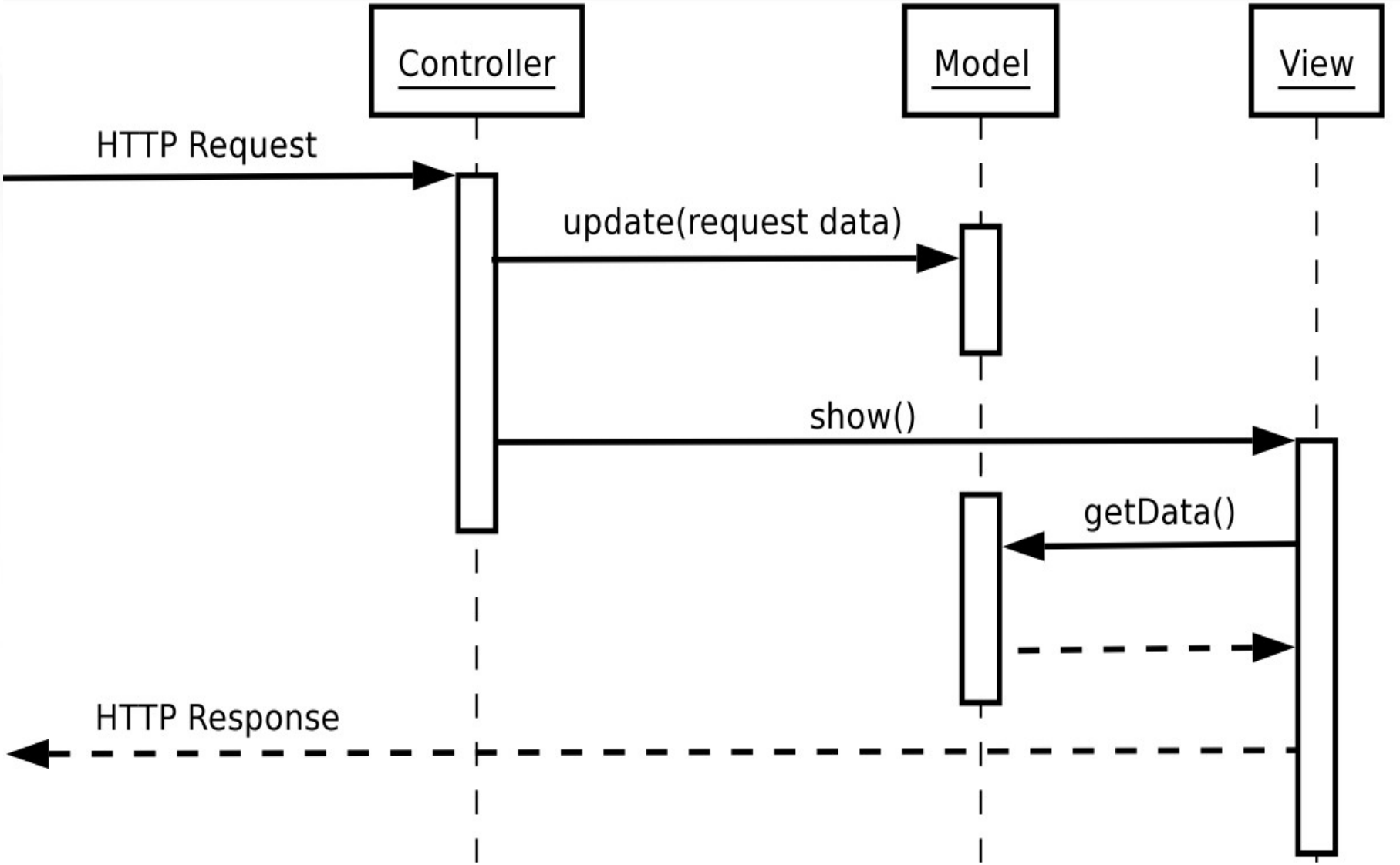
Харагдац (View)

- Харуулах элементүүдийг агуулна
- Нэг моделийг олон харагдац хэрэглэж болно
- Моделийн өөрчлөлтөд хариу өгнө
- Үзэгдлийг боловсруулахгүйгээр удирдлага уруу дамжуулна

Удирдлага (Controller)

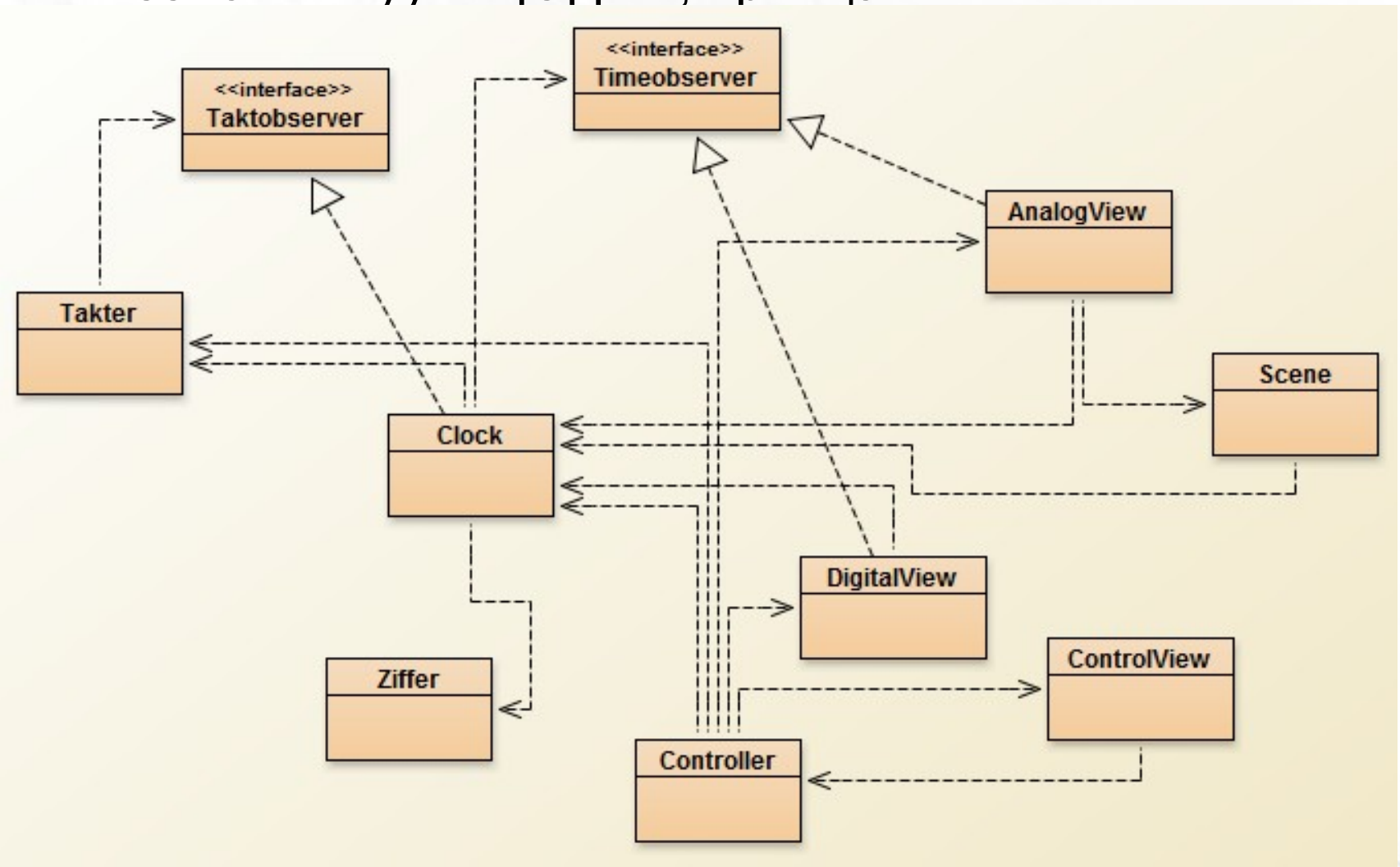
- Хэрэглэгчийн оруулсан өгөгдлийг хүлээн авч моделийн зурвасын функцийг хүлээн дуудна
- Заримдаа харагдацын дүрслэлийг өөрчилдөг
- GUI системүүдийн үзэгдлийг боловсруулна (Харагдацаас хүлээн авсан)

MVC SD

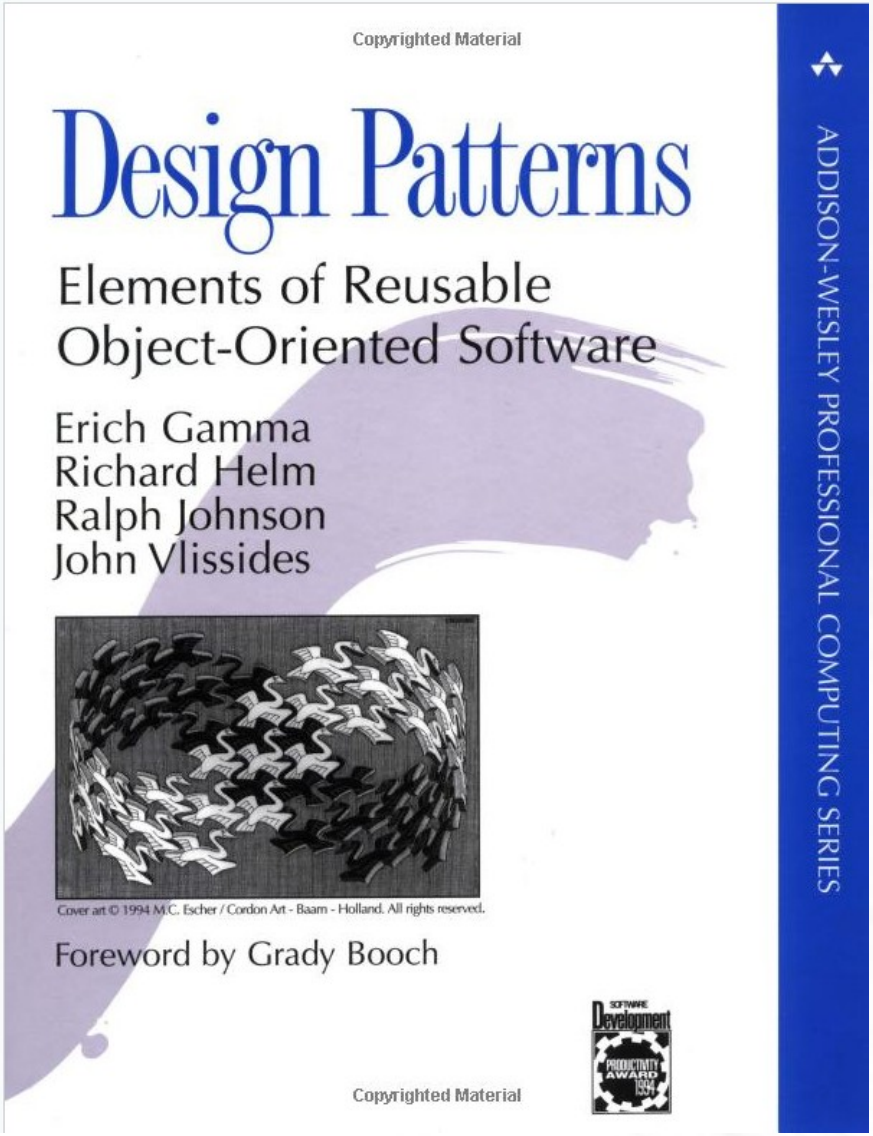

Controller
Model
View
HTTP Request
update(request data)
show()
getData()
HTTP Response

MVC жишээ

Жишээг ажиллуулж үзүүлж, ярилцав.



Судлах материал



Design Patterns: Elements of Reusable Object-Oriented Software

Erich Gamma,
Richard Helm,
Ralph Johnson,
John Vlissides

Дүгнэлт

- Зохиомжийн загвар гэж юу болох
- Хэрхэн ангилагддаг
- Яагаад судлах хэрэгтэй гэж
- Үүсгэгч загварууд – Abstract factory pattern, builder pattern, singleton pattern