# SE302 – ПХ БҮТЭЭЛТ

# Лекц 11

# Integration

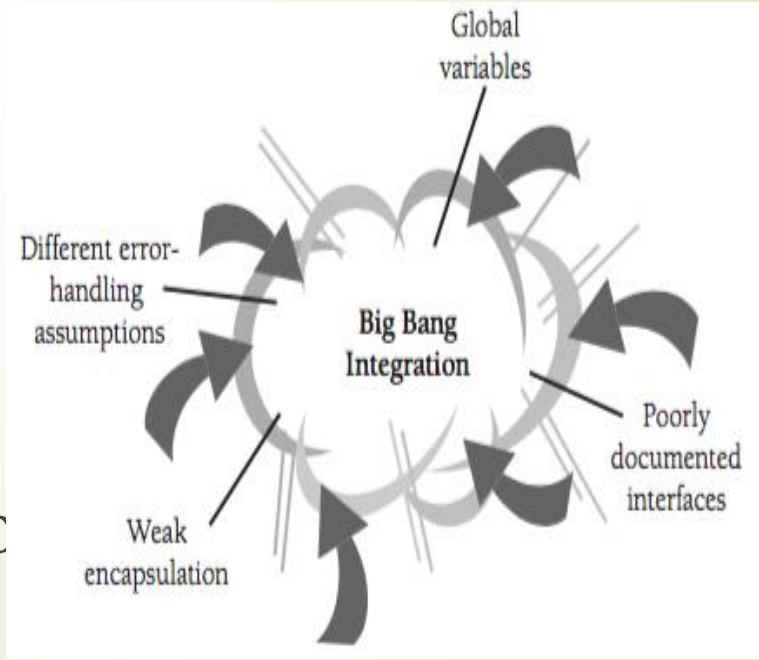
- Интеграцийн чухал хандлагууд
  - ПХ-аас өөр салбарт интеграци ямар чухал гэдгийг мэднэ
  - Washington-ний их сургуулийн хөлбөмбөгийн стадион

# Integration

- Phased Integration
  - Олон жилийн өмнө стандарт байсан
    - Дизайн, Кодчилол, Тестчилэл, класс бүрийг дебаг хийх
    - Классуудыг зохицуулж нэг том систем болгох (Системийн интеграци)
    - Системийг бүхлээр нь тестчилэх, дебаг хийх
  - Нэг алдаа нь
    - Систем дэх классууд анх удаа хамт ажиллах үед шинэ асуудал зайлшгүй гардаг
      - Шалтгаан нь хаана ч байж болно
      - Классуудыг муу тест хийсэн, классуудын интерфэйсийн алдаа, хамтын ажиллагааны алдаа гэх мэт

# Integration

- Phased Integration
  - Өөрөөр “Big Bang Integration” гэж нэрлэдэг
    - Их тэсрэлт
    - Бүх класс хөгжүүлэгчийн тест хийгдтэл эхэлж чаддагагүй
  - Жижиг програмд
    - Шаардлагагүй
  - 2, 3 класстай үед
    - Хамгийн тохиромжто

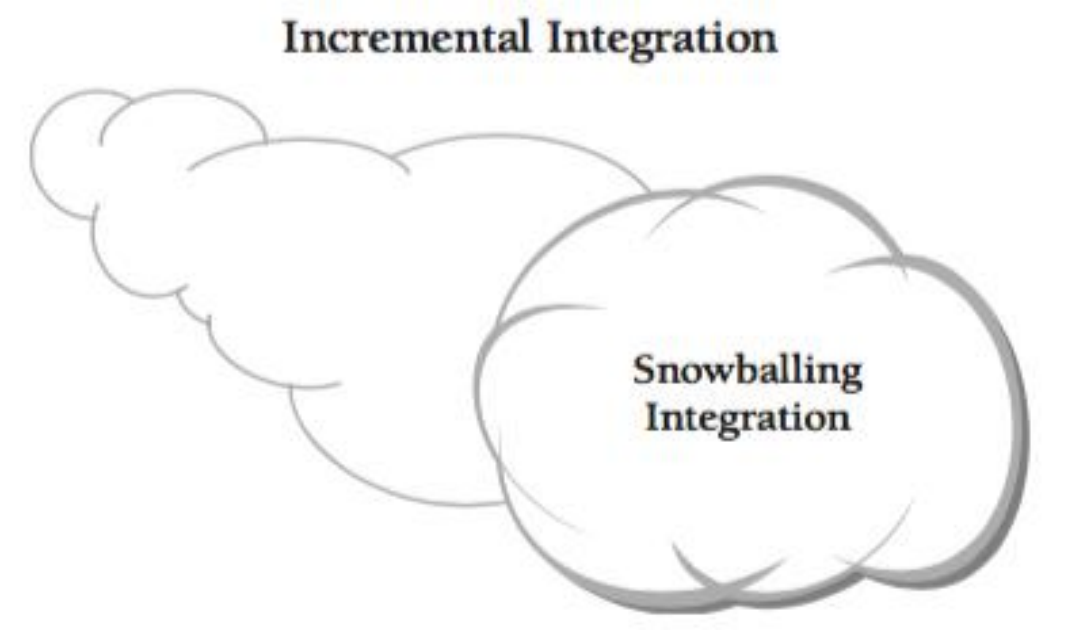


# Integration

- Incremental integration
  - Програмын жижиг хэсгийг бичээд, түүнийгээ зохицож байгаа эсэхийг тестлэнэ
  - Доорх алхамаар гүйцэтгэгдэнэ
    - Системийн жижиг хэсгийг хөгжүүлж, түүнийгээ тестлэнэ
    - Классын дизайн, кодчилол, тестчилэл
    - Системийн үндсэн араг ястай нэгтгэнэ (skeleton)
      - Түүнийг тестлэнэ
      - Шинэ класс нэмэхээс өмнө системийн араг яс хэвийн ажиллаж байгаа гэдэгт итгэлтэй байх ёстой

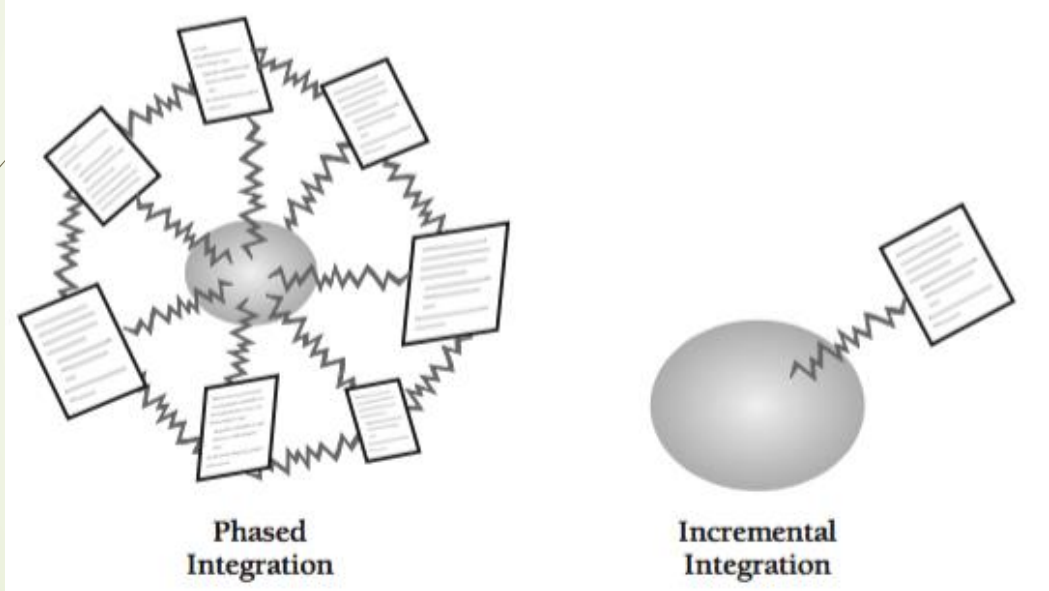
# Integration

- Incremental integration
  - Систем зогсолтгүй шинэчлэгдэж байхад интеграци хийхэд тусладаг



# Integration

- Incremental integration-ний давуу талууд
  - Алдааны байршлыг тодорхойлоход хялбар




  - Систем эрт зорилгодоо хүрнэ
    - Интеграци хийгээд ажиллахад, үр дүнг ашиглахгүй ч програмистууд өөрсдийн ажлын үр дүнг эрт харна

# Integration

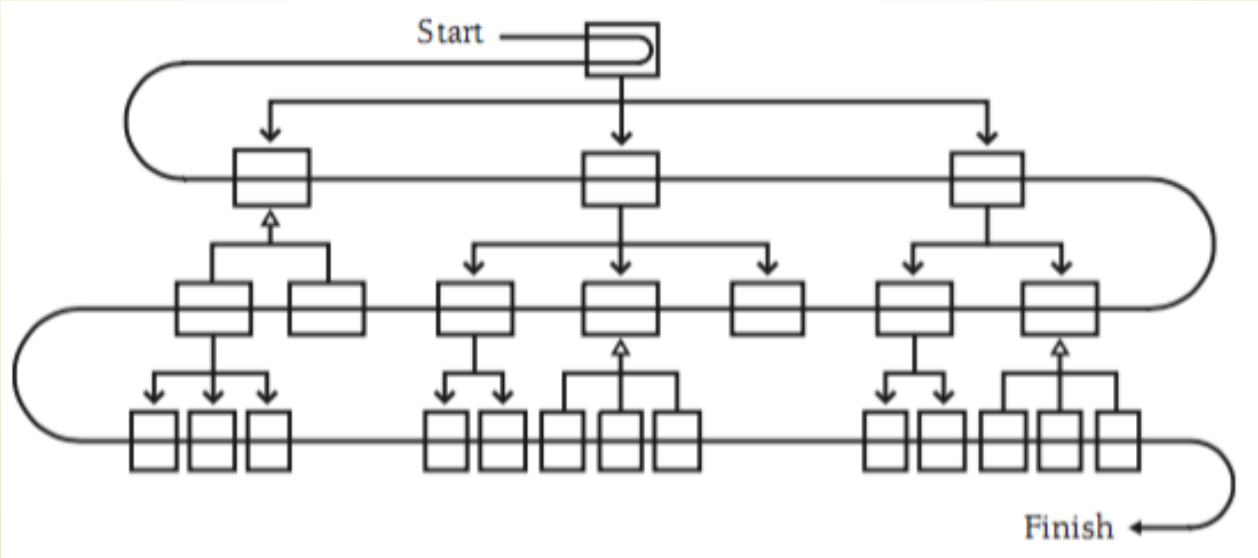
- Incremental integration-ний давуу талууд
  - Хяналтыг сайжруулна
    - Олон давтан интеграци хийхэд
      - менежментийн баг кодчилол 99% тай гүйцэт гэж сонссоноос 50%-тай ажиллаж байгааг харах нь илүү дээр
  - Үйлчлүүлэгчийн харилцааг сайжруулна
  - Системийн нэгж хэсгүүд илүү гүйцэт тестлэгдэнэ
    - Төсөлд интеграци эрт эхэлнэ
    - Аль ч тохиолдолд классад хөгжүүлэгчийн тест хийнэ
  - Системийг богино хугацаанд хөгжүүлж чадна
    - Интеграцийг сайн төлөвлөвөл
      - Системийн нэг хэсгийн дизайн гаргаж байхад, өөр хэсгийг кодлож болно
      - Энэ нь ажиллах цагийг бууруулахгүй ч, зэрэгцээ явуулна

# Integration

- Incremental integration-ний стратеги
  - Шаталсан интеграцийн үед
    - Аль програмын компонентууд угсрах гэдэг дараалал байхгүй
    - Бүх компонентууд зэрэг интеграци хийгдэнэ
  - Incremental integration-ний үед
    - Төлөвлөгдсөн байна
    - Бусад компоненттэй интеграци хийхээс өмнө
      - Систем тэднийг дуудсан байна (холбогдсон)

# Integration

- Top-Down Integration
  - Шаталсан бүтцийн дээгүүр байрлах классууд эхэлж бичигдэх бөгөөд интеграци хийгдэнэ
  - Дээр байгаа нь
    - Main window, програмын control loop, JAVA-ийн main() агуулсан объект



# Integration

- Top-Down Integration
  - **Stubs** буюу хүү салбар классууд эцэг классыг дасгалжуулах байдлаар бичигдэнэ.
  - Классууд дээрээс доош зарчмаар нэгтгэгдэнэ
    - Stub-ууд бодит классаар солигдоно
  - Чухал зүйл нь
    - Класс хоорондын интерфэйс нь маш анхааралтай тодорхойлогдсон байх
    - Түгээмэл гардаг асуудал нь зөвхөн нэг классад нөлөөлдөггүй
      - Класс хоорондын нарийн харилцаанаас үүсдэг
    - Маш сайн тодорхойлсон интерфэйс нь асуудлыг бууруулна
      - Интерфэйс нь интеграци хийх үйл ажиллагаа биш

# Integration

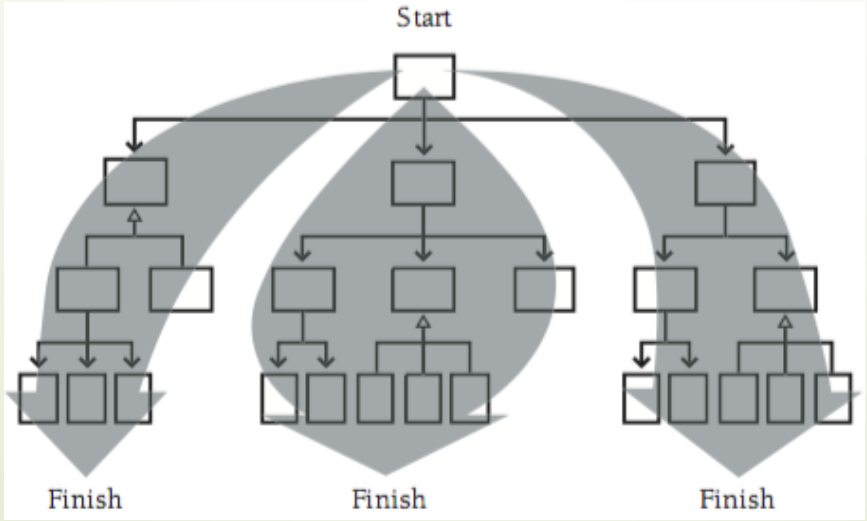
- Top-Down Integration
  - Бусад incremental integration-с давуу тал нь
    - Системийн контрол логик нь харьцангуй эрт тестлэгддэг
    - Шатлалын дээгүүр байрлах классууд
      - Том
      - Алсыг харсан
      - Дизайны асуудлууд нь ил гардаг
  - Өөр давуу тал нь
    - Хянамгай төлөвлөж чадвал
      - Төсөлд системийг хэсэгчлэн эрт ажиллуулж чадна
      - UI хэсэг дээгүүр байрлаж байвал анхдагч UI-ийг эрт үзэж, бага багаар нарийн хэсгүүдийг нэмж чадна
    - Хэрэглэгч болон програмистын ойлголцлыг нэмэгдүүлнэ
      - UI-д үзэгдэх зарим зүйлсийг эрт илрүүлнэ

# Integration

- Top-Down Integration
  - Кодчилол эхлэхээс өмнө доод түвшний нарийн дизайнууд гүйцсэн байх ёстой
  - Системийн интерфэйсүүд нь алдаатай эсвэл гүйцэтгэлийн асуудалтай бол түүнд хүрэх гэж урт зам туулна
    - Төсөл дуусахын өмнөхөн
    - Энэ нь зөвхөн доод түвшний асуудал биш
    - Системийн дээд түвшинд асуудал хүрч болно
    - Дээд түвшний өөрчлөлтийн шалтгаан нь эрт интеграци хийсний давуу талыг бууруулна

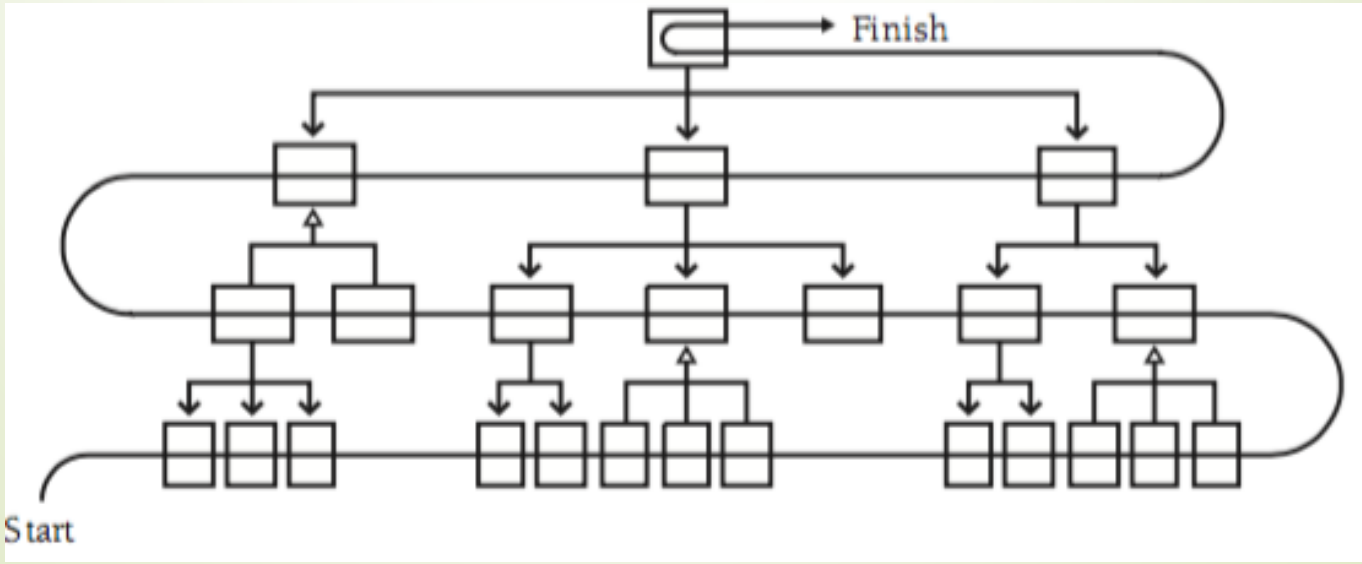
# Integration

- Top-Down Integration
  - Классын цуглуулгууд нь дээрээ байрладаггүй бол энэ аргыг ашиглах боломжгүй
    - Ихэнх системүүдийн хувьд UI нь дээрээ байрладаг
    - Бусад системүүдийн хувьд main() функц
  - Өөрөөр босоо зүсэх арга



# Integration

- Bottom-Up Integration
  - Классуудыг доороос нь дээш бичиж нэгтгэнэ
  - Доод түвшний классуудыг бичнэ
  - Тэдгээрийг дасгалжуулах test-driver бичнэ
    - Test-driver руу доод түвшний классуудыг нэмнэ
  - Test-driver-ийг бодит дээд түвшний классаар солино

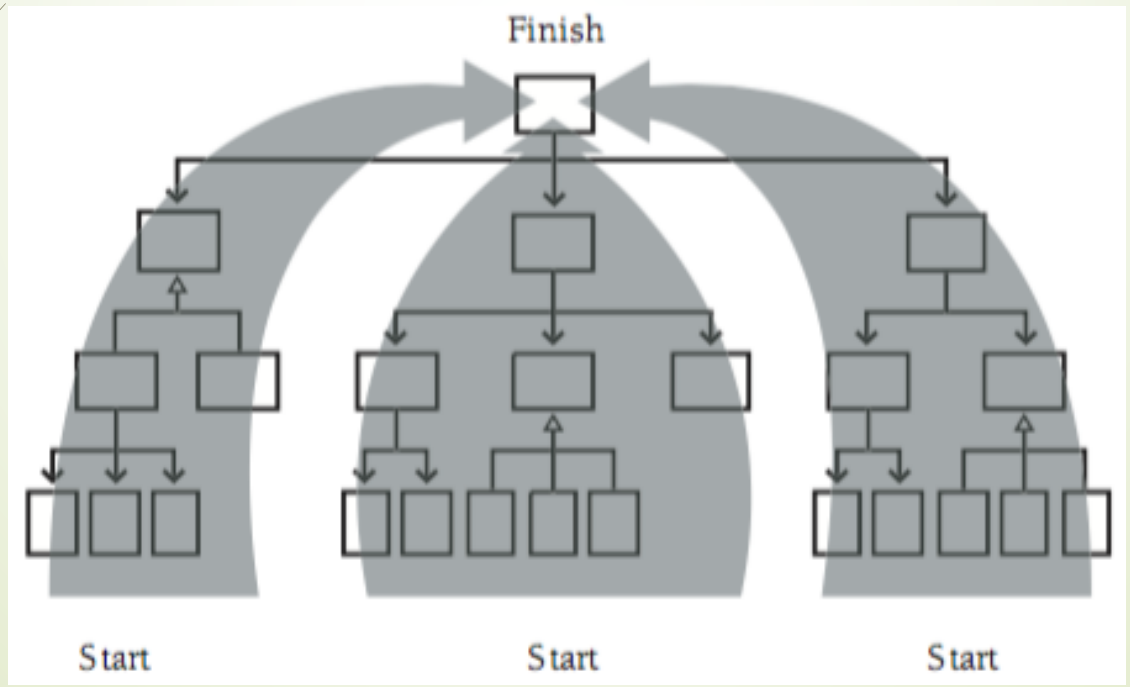


# Integration

- Bottom-Up Integration
  - Хязгаарлагдмал давуу талыг олгодог
    - Класс нэгтгэх үед алдаа олоход хялбар
    - Интеграцийг эрт эхлүүлдэг
      - Системийн интерфэйсийг эрт дасгалжуулдаг
  - Энэ аргын гол асуудал нь
    - Гол интеграцийг орхигдуулдаг
    - Дээд түвшний системийн интерфэйс хамгийн сүүлд
    - Систем нь дээд түвшиндээ дизайны асуудал тулгарвал
      - Доод түвшинд аль хэдийнэ хийгдсэн зарим ажил цуцлагдана
    - Интеграцийг хийхээс өмнө системийн бүтэн дизайныг шаарддаг

# Integration

- Bottom-Up Integration
  - Top-down шиг дангаар нь ашиглагдах нь ховор, оронд нь эрлийз аргыг ашиглах

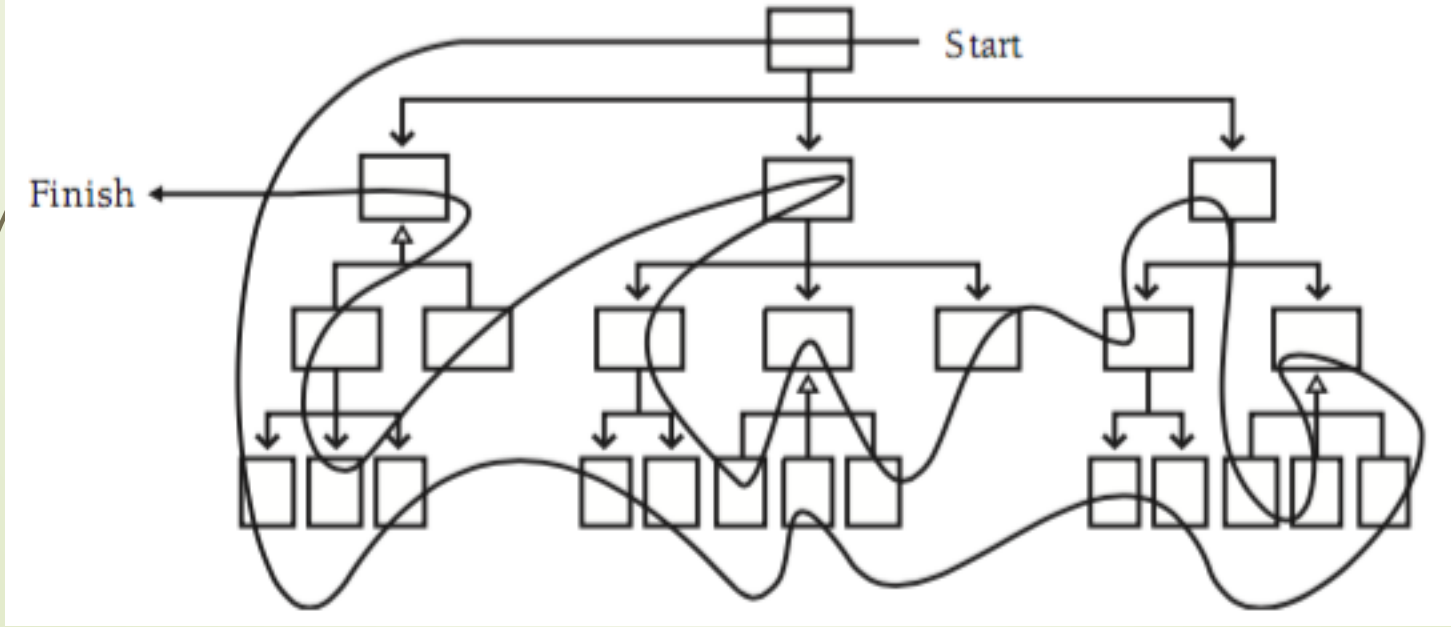


# Integration

- Sandwich Integration
  - Цэвэр дээрээс доош, цэвэр дооройс дээш аргын оронд экспертүүд сэндвич интеграцийг ашиглахыг зөвлөдөг
    - Эхлээд шатлалын дээр байрлах дээд түвшний бизнес классуудыг тодорхойлно
    - Тэгээд доор байрлах төхөөрөмжийн интерфэйс класс болон өргөн хэрэглэгддэг utility классуудуудтай интеграци хийнэ
    - Эдгээр дээд түвшний болон доод түвшний классууд нь сэндвичийн талх юм.

# Integration

- Sandwich Integration
  - Дунд түвшний классуудыг орхино
    - Мах, бяслаг, помидор
  - Бодитой практик арга барил

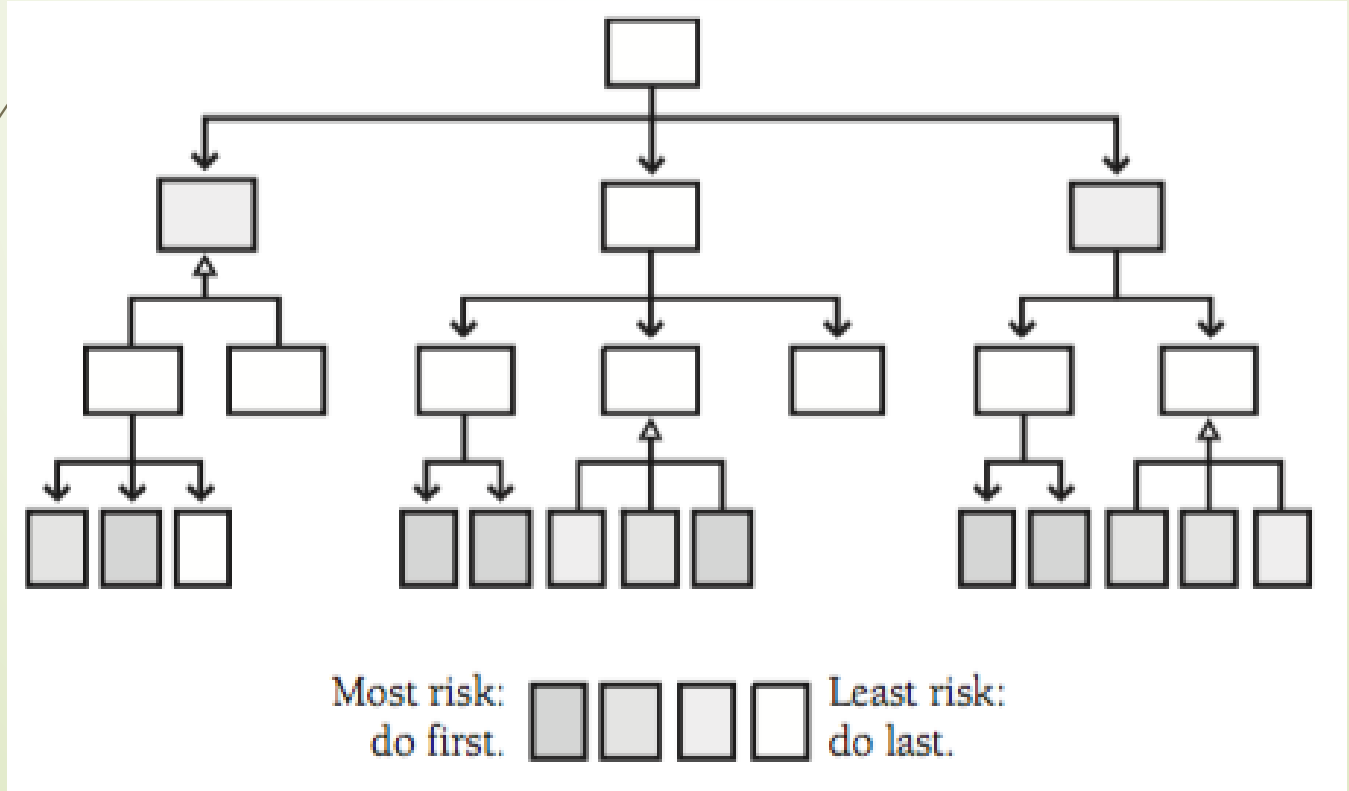


# Integration

- Risk-Oriented Integration
  - Өөрөөр “hard part first integration” гэж нэрлэдэг
  - Цэвэр дооройоос дээш, дээрээс доош интеграцын асуудлаас зайлсхийсэн
    - Сэндвич интеграцитай төстэй
    - Дээд болон доод классууд эхэлж интеграци хийгдээд, сүүлд нь дунд байрлах классууд
  - Эрсдлийн хандлагат интеграцид класс бүрийн эрсдлийг тодорхойлно
    - Хэрэгжүүлэхэд хамгийн хэцүү байж болох хэсгүүдийг эхэлж хэрэгжүүлнэ
    - Туршлагаас харахад
      - Системийн дээд түвшний интерфэйсүүд эрсдэлтэй учираас эхэнд, мөн шатлалын доод түвшинд эрсдэлтэй байдаг

# Integration

- Risk-Oriented Integration
  - Хамгийн эрсдэлтэй классуудыг эхэлж интеграци хийгээд дараа нь хялбар классуудыг

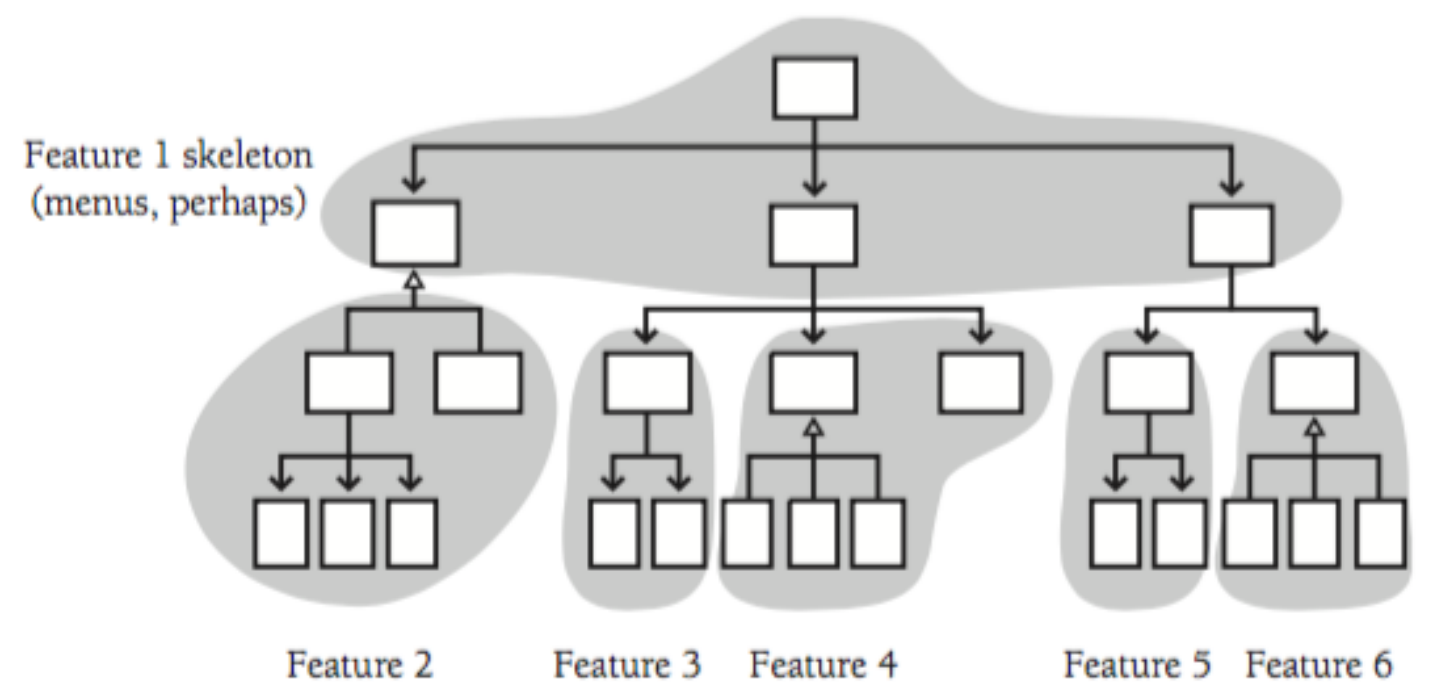


# Integration

- Feature-Oriented Integration
  - Нэг удаад нэг онцлогийг тусгах арга юм.
    - Текст боловсруулах програм бичиж байгаа бол underline-ийг дүрслэх, документыг автоматаар форматжуулах гэх мэт
  - Ихэвчлэн бусад боломжуудыг дэмжих зорилгоор Skeleton-с эхэлдэг
    - Интерактив системийн хувьд онцлох зүйл нь интерактив меню байж болно

# Integration

- Feature-Oriented Integration
  - Онцгой зүйлсээс бүтсэн мод



# Integration

- T-Shaped Integration
  - Нэг босоо хэрчим нь эрт хөгжүүлэлт болон интеграци хийгддэг.