8. Модели на разпределеното обслужване

ВАСИЛ ГЕОРГИЕВ

v.georgiev@fmi.uni-sofia.bg

Модели на разпределено обслужване

- Сървърни разпределени услуги
  - Клиент-сървер
  - Трислоен модел
  - Брокерен модел
  - Сервизно-базиран модел
  - Компонентно-базиран модел (не)
- Безсървърно обслужване (p2p) – в 9. л-я

Обхват

- MPMD/MU – за множество потребители (Системи за масово обслужване)
- Характеристики
  - Топология – звезда, дърво/нелинеен конвейер, пълен граф, непълен граф
  - Грануларност
    - декомпозиция на
      - сървера: услуги, компоненти…
      - данните
  - Обмен – синхронен/асинхроннен/срочен (РВ)
  - Управление
    - по време/събитие
    - разпределено /централизирано

Случаи

- Разпределени бази данни, вкл.:
  - балансирана репликация на данните
- B2B (бизнес-бизнес) склад-каталог-доставчици – при алтернативи на услугите
  - образователни услуги; финансови услуги; транспортни услуги; здравни услуги; производители на изделия… – множество потребители и множество доставчици
- Разпределени ресурси – SAP, облачно обслужване с арендувани ресурси или федерирани в грид ресурси

Модели

- Клиент-сървер
- Многослоен сървер
- Брокерно-базирана архитектура – CORBA
- Сервизно-базирана архитектура – SOA
- Web-услуги, грид-услуги
  - характеристики на услугите
    - адресна прозрачност на услугата (location transparency)
    - откриваемост и режим на достъп (за грид – single sign-in)
    - надеждност
      - QoS-параметри, произтичащи от локална и мрежова производителност, моделът на натоварване, защита на данните и др.
    - наличност

Платформи

- Разпределени бази данни
- Балансирана репликация на данните
- B2B (бизнес-бизнес) склад-каталог-доставчици – при алтернативи на услугите
  - образователни услуги; финансови услуги; транспортни услуги; здравни услуги; проиводители на изделия… – множество потребители и множество доставчици
- Разпределени ресурси – SAP, облачно обслужване с арендувани ресурси или федерирани в грид ресурси

Клиент-сървер

- Класическият MPMD-модел на адаптиране на ограничената клиентска инфраструктура за изпълнение на «сложни» и съставни услуги – от тънък до богат клиент, граничещ с безсърверно обслужване (p2p) – активен (инициативен) интерфейсен процес и реактивен процес за приложната логика и данни (фиг. 8.7.)
- Предимства
  - технологично специализиране
  - инфраструктурна гъвкавост (клиентът е с достъп до интерфейси и сензори, сървърът е без такива, но с планирана и проектирана производителност и свързаност)
  - преизползване на сървърните компоненти
  - еволюция на услугите без участието на потребителя
- Недостатъци
  - унификация на клиентската инфраструктура (напр. браузерен достъп)
  - защита на достъпа и данните
  - скалируемост на сървера
  - скъпа поддръжа, профилактика, тестване

Многослоен сървер

- Декомпозиция на сървера минимум на два слоя (фиг. 8.8.)
  - междинен слой (между клиента – интерфейсния слой и вътрешния слой) за приложната (“business”) логика
  - вътрешен слой – обикновено за достъпа до данни или комуникации
- Предимства на междинния слой
  - технологично специализиране
    - напр. клиентът не генерира SQL-заявки и е напълно прозрачен за сървърната имплементация
  - инфраструкторна гъвкавост
    - напр. изтегляне и независимост на трафика от [ненадеждната] клиентска инфраструктура
  - преизползване на сървърните компоненти
  - еволюция на услугите без [сложно] да се еволюират данните – напр.:
    - многонишкова междинна услуга за ускорение или скалируемост
    - рекурсивен сървер с нова нишка за всяка нова заявка и/ли потребител като Apache –
    - итеративен сървер с изпълнение на конкурентните заявки една по една от опашка като Xampp – практичен при малки и балансирани натоварвания
- Недостатъци
  - унификация на клиентската инфраструктура (напр. браузерен достъп)
  - защита на достъпа и данните
  - скалируемост на сървера
  - поддръжка, профилактика, тестване – в още по-голяма степен

Брокер на услуги

- Брокерът е инстанция на междинния слой, която поддържа достъпът до и обработката на множество услуги или алтернатвни услуги или външни услуги (фиг. 8.9.) чрез:
  - à la директория на услугите
  - контекстно (т.е. приложно) класиране на услугите – функционално, а и нефункционално
  - динамично адресиране (прозрачно, по параметър – look-up) освен статичното при [3t]C-S (по адрес или име, но които са с уникалност на идентификатор)
- Функции
  - регистриране на услугите и актуализация на регистъра
    - обикновено с публикуване на заредим от клиента интерфейс на услугите (напр. бисквити и проксита)
  - достъп на клиентските процеси
  - класиране и препращане на клиентските заявки към една или повече услуги
  - връщане на генериран резултат или обработка на изключението
- Обменът се базира на клиентско и брокерно прокси – допълнителен протокол (т.е. резидентен процес), обикновено изпълняващ функциите на [де-]сериализиране ([un-]marshalling) и други представителни преобразовния

Брокерна и междуброкерна архитектура

- Архитектура (sd) – фиг. 8.10.1
- Междуброкерни мостове – фиг. 8.10.2
  - имат функциите на „шлюз“ от високо ниво – преформатират съобщенията между брокери с различна технология
    - DCOM
    - .NET Remote
    - Java CORBA
- Недостатъци на брокерната архитектура
  - свръхтовар и поддръжка
  - множеството административни области в сървърната страна – необх. преформатиране на заявките и потенциал за грешки

Брокерни инстанции – CORBA (Common object request broker architecture)

- Обектно-базирана стандартна брокерна платформа с RMI – брокерът е
  - само транспортен междинен процес за езикова независимост и прозрачност между клиента и услугата (за наследени и хетерогенни сървърни приложения) адресна прозрачност но без директория и класиране на услугите
- статично обръщение в CORBA
  - интерфейсна спецификация на IDL (Interface Definition Language) със статична генерация на клиентски и сървърен („skeleton”) стъб – фиг. 8.11;
  - стъбовете и скелетоните се генерират статично от IDL-компилатор
  - генерираните стъбове се свързват статично (link) с клиентския и сървърния код

Динамично обръщение в CORBA

- DII (Dynamic Invocation Interface) – базира се на динамично зареждане на клиентския стъб по време на изпълнение и то при възникнала заявка:
  - скелетонът резидира в сървърната страна, а клиентският стъб е депозиран в интерфейсен склад – IFR (CORBA Interface Repository)
  - клиентския стъб се зарежда в клиентската страна от IFR чрез ORB (Object Request Broker), който е протокол за обръщение и зареждане – фиг. 8.12

Системи с брокери на съобщения – фиг. 8.13

- Базират се на [XML-]съобщения за обмен на форматирани данни между приложенията
- прилагат се при слабо-свързани приложния или системни процеси – т.е. при някои от следните характеристики:
  - с неявна адресация – на базата на аргументи в самото съобщение
    - напр. „файл за печат на цветен принтер на 3 етаж с най-къса опашка за задания в момента“
  - с повече от една възможни инстанции-реплики
  - в 2+ административни области, вкл. изпълняващи различни защитни протоколи
  - йерархични но и равнопоставени комуникационни модели, т.е. освен 1:1, 1:* също pub-sub т.е. тематична адресация
- поддържат се от всички доставчици – WebSphere, IBM MQSeries, JMS.

👍 преизползване, преносимост, административна независимост, интегриране на QoS-черти на база на текущ/динамичен и/ли отдалечен контекст; управление по събитие чрез кодиране и интерпретиране („парсване“) на съобщенията

👎 свръхтовар, трудна настройка и тестване

SOA

- Сервизно-базираните архитектури са слабо-свързана едро-грануларна платформа за множествено обслужване с декомпозиция на ниво Услуга
- услугата е (фиг. 8.14)
  - завършена бизнес-функция: типично е конвейер, започващ със системна функция от диаграмата на случаите на употреба, композирана с последователност от системни, междусистемни и извънсистемни дейности, описани с диаграма на дейностите – напр. потребител <чекира> изделие, което последователно се кредитира, резервира в склада, опакова, предава на куриер и т.н. – не непременно само ИТ-базирани дейности
  - самостоятелен независим от други услуги процес (освен по данни – например складова наличност)
  - достъпен само чрез своя публикуван стандартен интерфейс – в стандартна директория на услугите
- гъвкавост, разширимост, преизползване, еволюция, наследен код в интерфейсна обвивка (wrapper)
- обменът е чрез стандартни протоколи – SOAP и WSDL

SOA – композиция

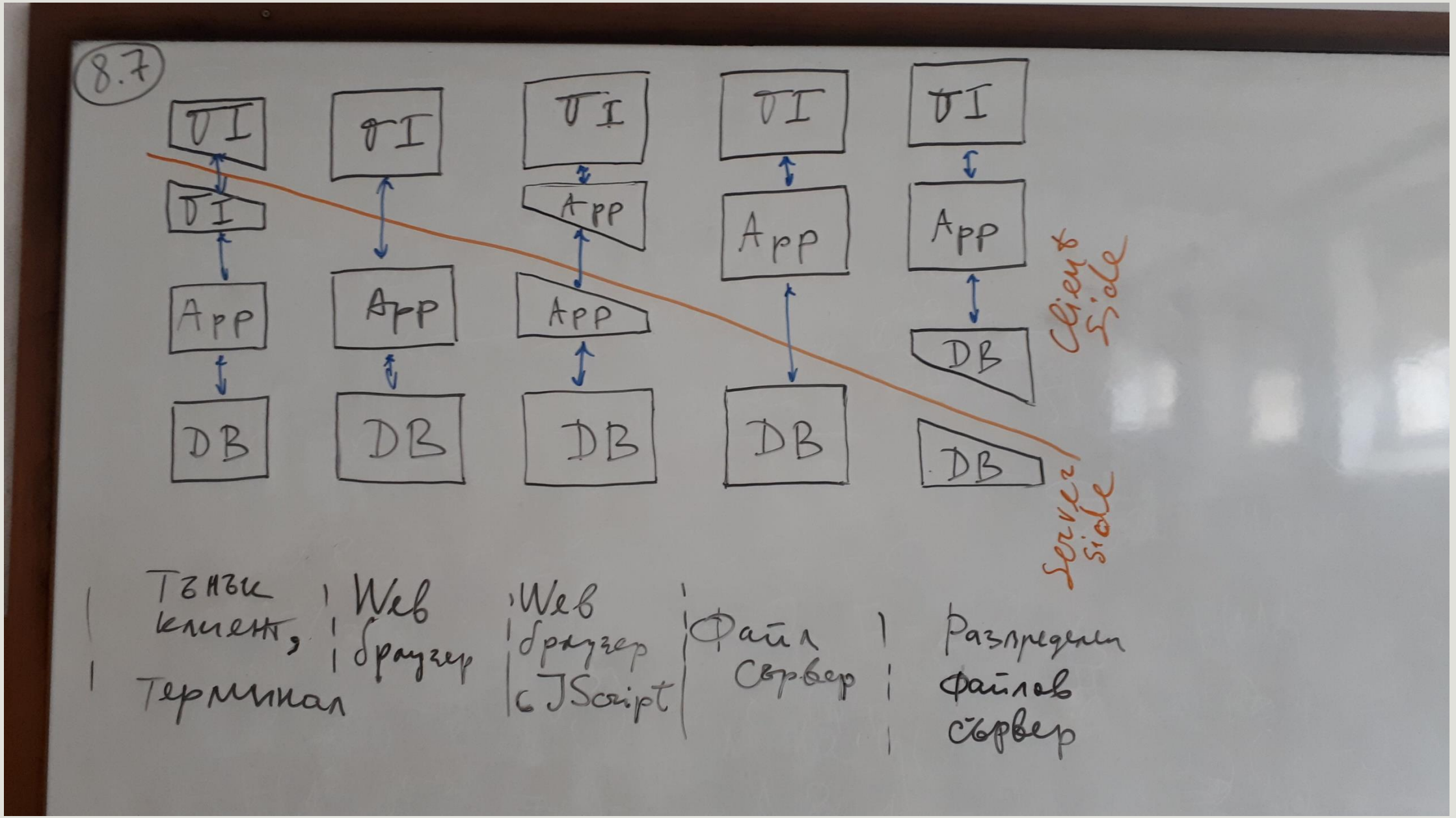
- Изискването за завършеност се имплементира чрез йерархична композиция на услуга от „суб-“услуги – подреждане на инициативата на обръщение от супер-услугата към суб-услугите – типично атомарни (неделими и завършени) услуги с Web-интерфейс
- Пример за композирана услуга в система за електронна търговия – фиг. 8.15.1
- Композираната услуга наследява прозрачно (seamless) еволюция в чертите на отделните атомарни услуги
- атомарните услуги могат да се преизползват в няколко бизнес-услуги, които използват еднакъв интерфейс на достъп
- атомарните услуги като правило не са публикувани услуги или ако са – те представляват сами по себе си бизнес услуга
- композицията може да е на няколко йерархични нива – фиг. 8.15.2

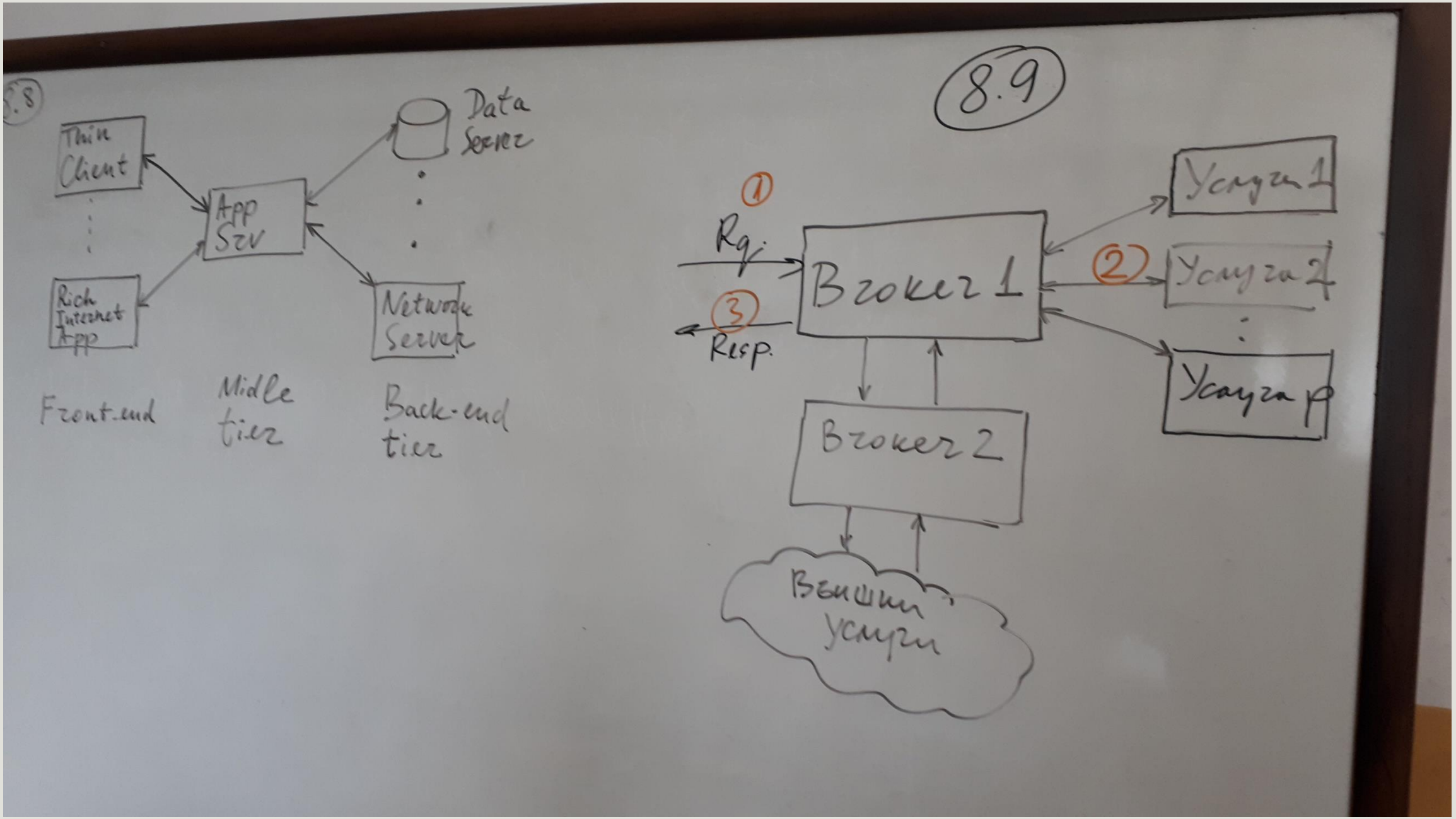
SOA – предимства

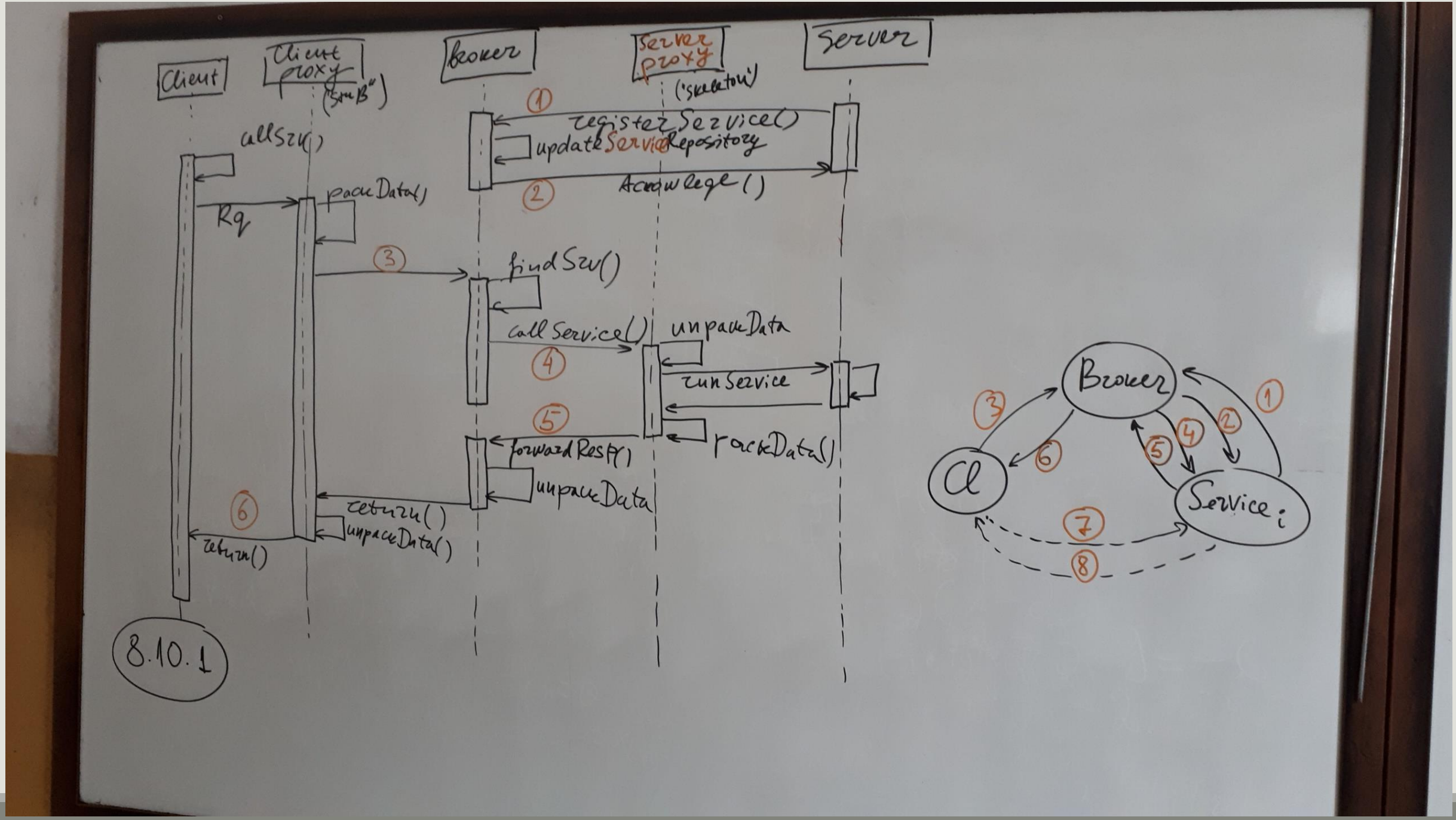
- слабосвързани т.е. независими компоненти – възможност за прозрачна локална еволюция на всяка атомарна услуга
- платформена и езикова независимост – при използване на стандартни интерфейсни протоколи
- преизползване на различни нива – публикувани и непубликувани атомарни услуги
- обменът на съобщения се структурира като асинхронен документен обмен
- висока ефективност по развойно време и себестойност (ниска ресурсна ефективност като всяко едро-грануларна обработка) –
  - за композирането на нов клиент е достатъчно да се зареди публичния интерфейс на услугата
  - сървърът на директорията на услугата не е тясно място, тъй като той сáмо връща публикувания сервизен интерфейс – фиг. 8.16

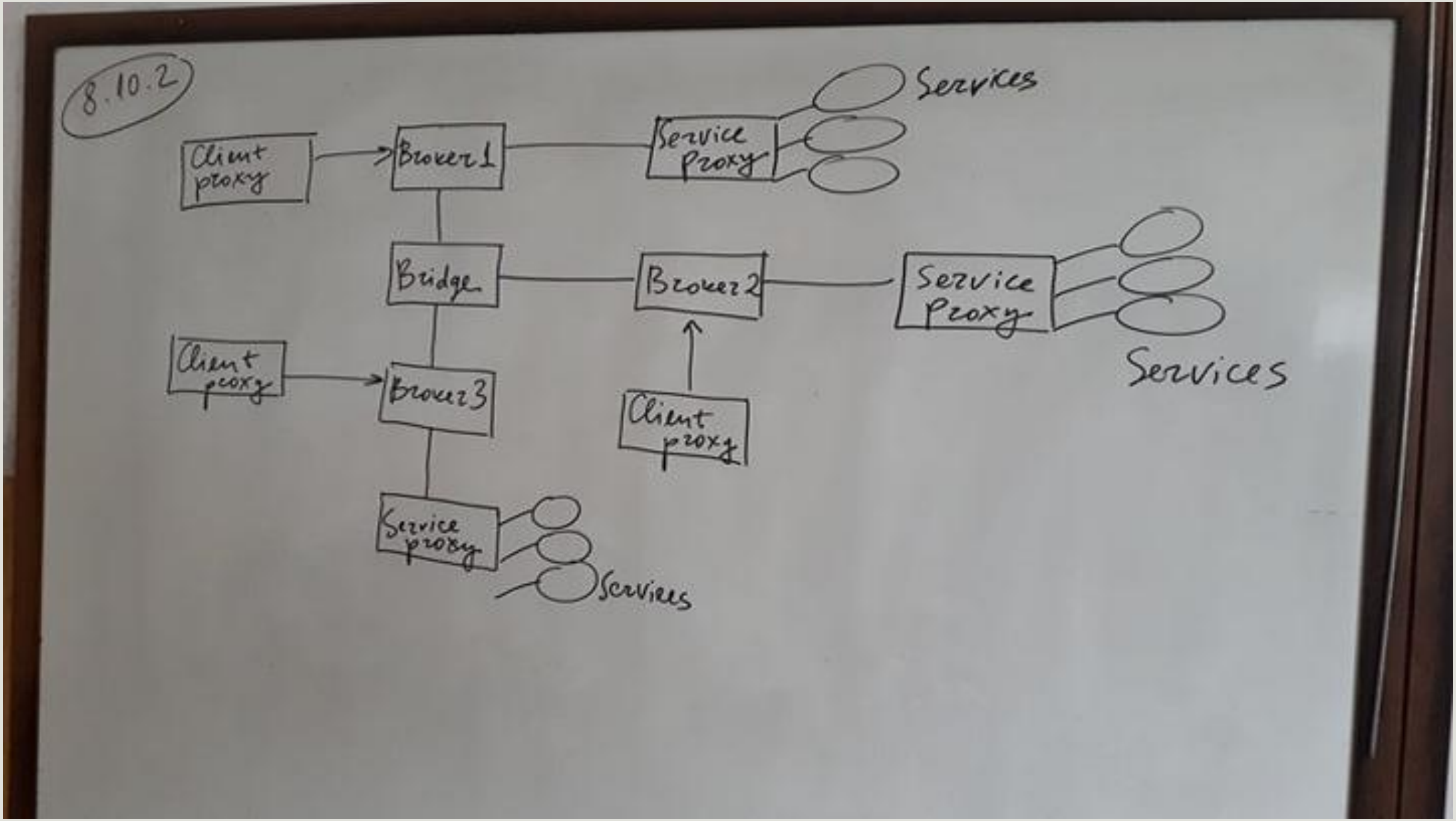
SOA – имплементация

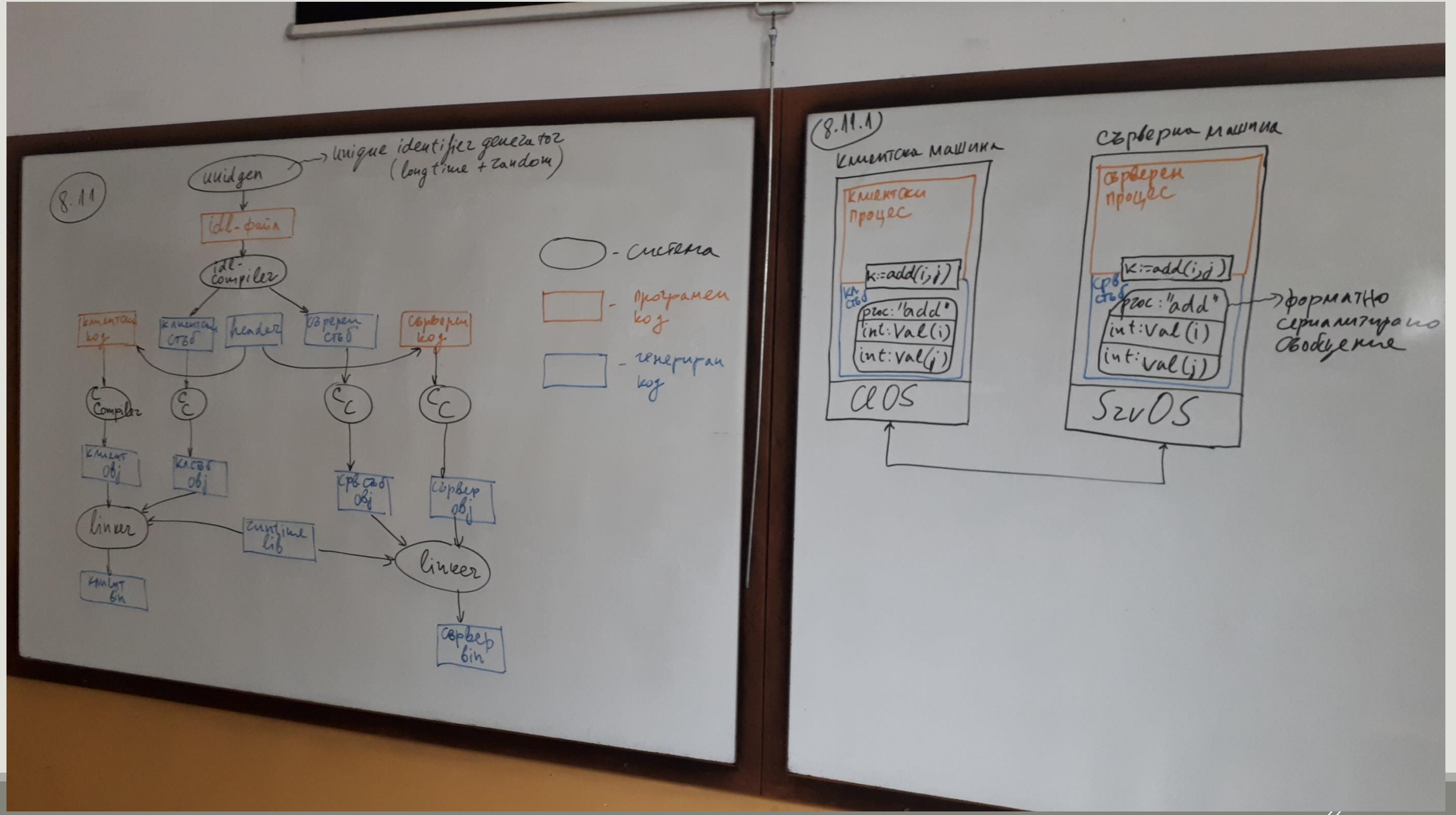
- Базира се на Web-услуги:
  - описани за публикувана с WSDL-записи/документи
  - достъпни за клиентите и за обмен при сервизната композиция чрез SOAP (Simple Object Access Protocol, алтернативи RPC, RMI) – фиг. 8.17
  - преносими поради браузерния достъп, но ако са със собствено клиентско приложение съдържанието на съобщенията винаги се пакетира като HTTP-съобщение (рядка алтернатива SMTP) поради
    - документния му тип
    - проницаемостта на защитните стени
- SOAP е необходима вътрешна обвивка на съобщенията, тъй като
  - задава адресирането чрез име на услугата в регулирано пространство на имената
  - други атрибути – защитен протокол, издател и получател на съобщението, тип на съдържанието, мета атрибути (потребител, време, срок)
- Тялото на съобщението е XML-базирано съдържание

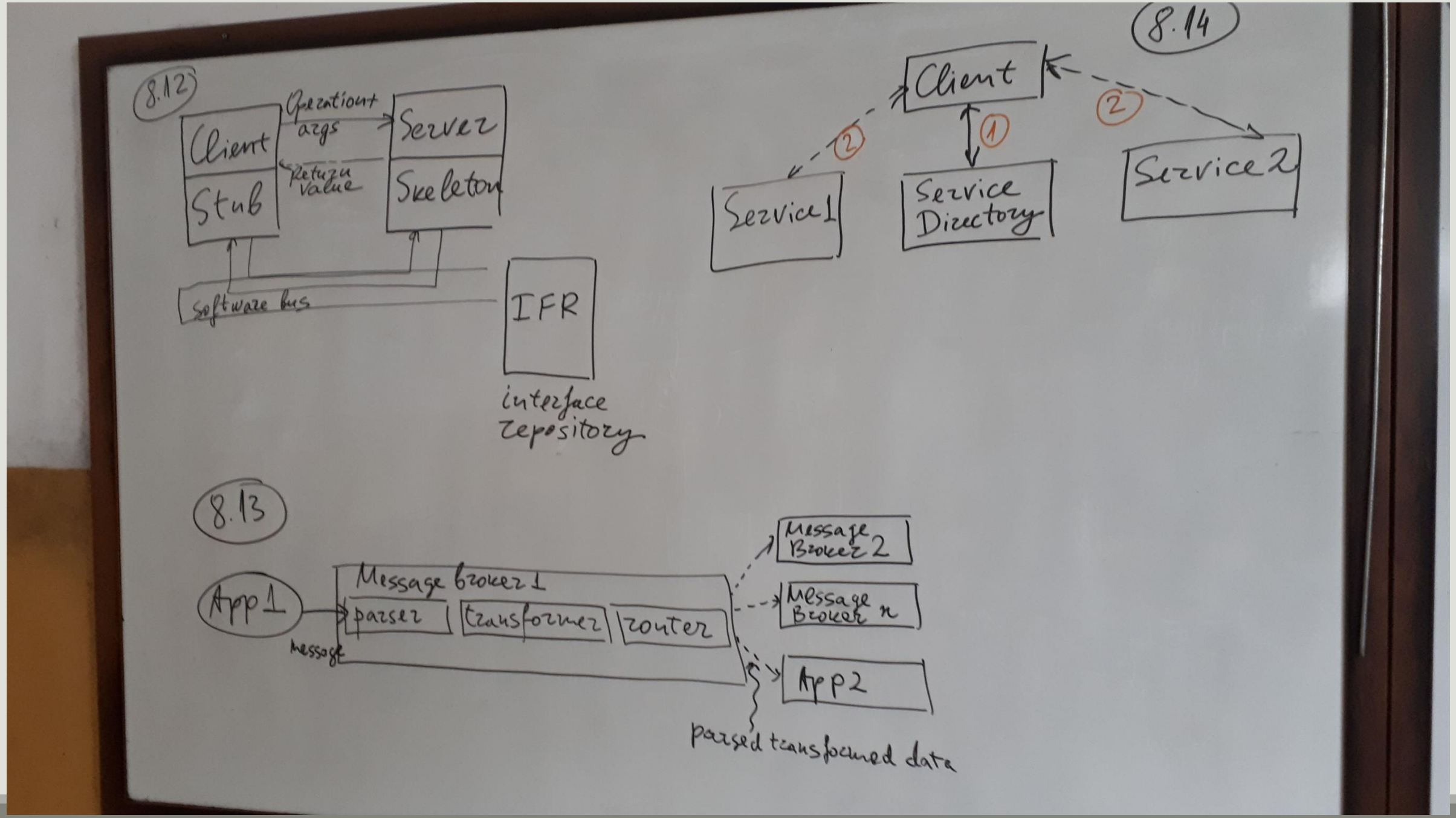

8.7
UI
UI
App
DB
UI
App
DB
UI
App
App
DB
UI
App
DB
UI
App
DB
DB
Client side
Server side
Тонкий клиент, терминал
Web браузер
Web браузер с JScript
Файл сервер
Распределенный файловый сервер

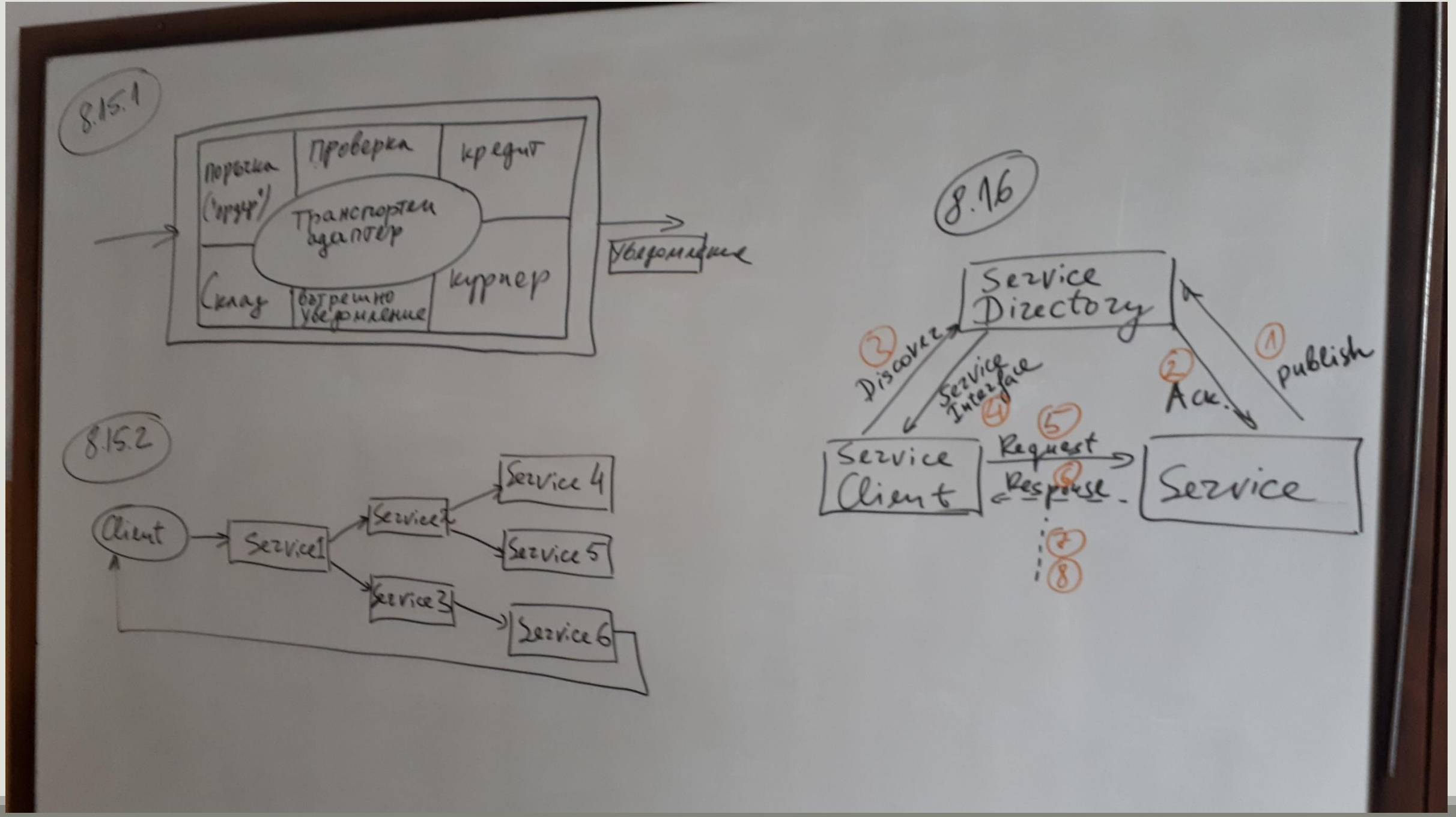

8.8
Thin Client
Rich Internet App
App Srv
Data Server
Network Server
Front-end
Midle tier
Back-end tier
8.9
1
Rq
3
Resp.
Broker 1
2
Broker 2
Външни услуги
Услуга 1
Услуга 2
Услуга р


Client
Client proxy
("stubB")
Broker
Server proxy
("skeleton")
Server
callSzv()
packData()
Rq
registerSzvice()
updateServiceRepository
Acknowlege()
findSzv()
callService()
unpackData
runService
packData()
forwardResp()
unpackData
return()
unpackData()
return()
8.10.1
Broker
Cl
Service i


8.10.2
Client proxy
Broker 1
Service Proxy
Services
Bridge
Broker 2
Service Proxy
Services
Client proxy
Broker 3
Client proxy
Service proxy
Services


8.11
uuidgen
Unique identifier generator
(longtime + random)
idl-файл
idl-compiler
клиентски код
клиентски стъб
header
сървърен стъб
сървърен код
C Compiler
CC
CC
CC
клиент obj
клстъб obj
срв стъб obj
сървър obj
linker
runtime lib
linker
клиент bin
сървър bin
- система
- програмен код
- генериран код
8.11.1
Клиентска машина
Сървърна машина
Клиентски процес
Сървърен процес
k:=add(i,j)
proc:"add"
int:val(i)
int:val(j)
ClOS
SzvOS
форматно сериализирано съобщение


8.12
Client
Stub
Operation + args
Return value
Server
Skeleton
software bus
IFR
interface repository
8.14
Client
1
2
2
Service 1
Service Directory
Service 2
8.13
App 1
message
Message broker 1
parser
transformer
router
Message Broker 2
Message Broker n
App 2
parsed transformed data


8.15.1
Поръчка ("заявка")
Проверка
кредит
Транспортен адаптер
Склад
Евтрешно уведомление
куриер
Уведомление
8.15.2
Client
Service1
Service2
Service3
Service 4
Service 5
Service 6
8.16
Service Directory
Discover
Service Interface
Ack.
publish
Service Client
Request
Response
Service