ПОНЯТИЕ ЗА СОФТУЕРНА АРХИТЕКТУРА

Какво е Софтуерна Архитектура (СА)

- Обикновено СА се създава като първа стъпка по време на проектирането, като целта е да се гарантира наличието на дадени качества в системата
- Детайли като алгоритми, представяне на данни, реализация, и т.н. не са предмет на СА
- Предмет на СА е поведението и връзките между различни елементи, разглеждани като “черни кутии”

Дефиниция

- Съгласно Software Engineering Institute:

  “Архитектура на дадена софтуерна система е съвкупност от *структури*, показващи различните софтуерни елементи на системата, външно видимите им свойства и връзките между тях”

Софтуерна архитектура

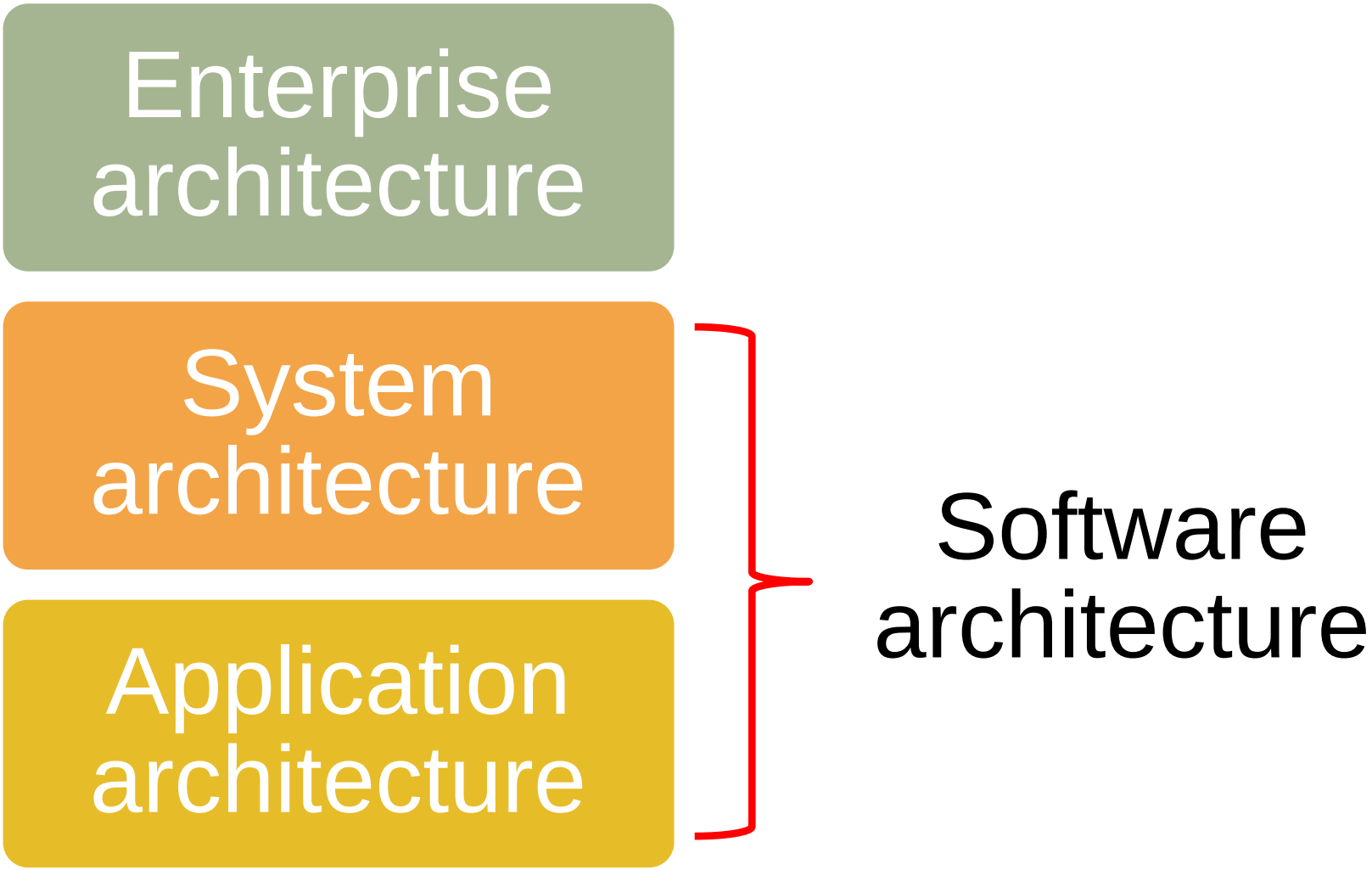
- Различни дефиниции
  - Основните решения по дизайна на софтуерната система, които включват
    - Структура
    - Поведение
    - Взаимодействия (вътрешни и с други системи)
    - Качествени характеристики
  - Подробно описание на конструкцията и начините за развитие на софтуерната система
  - И т.н.

- Над 50 дефиниции има на следния адрес:
  - http://www.sei.cmu.edu/architecture/start/glossary/community.cfm

По-общо понятие за архитектура

- Организационна архитектура (Enterprise architecture)
  - Основните процеси, технологичните и бизнес-стратегии в дадена организация
- Системна архитектура (System architecture)
  - Организацията на програмите и инфраструктурата върху която те се изпълняват
- Архитектура на приложението (Application architecture)
  - Организация на приложение, подсистема или компонент

По-общо понятие за архитектура



Архитектурни структури

- Структура – съвкупност от софтуерни елементи, техните външно видими свойства и връзките между тях;
- Изглед (view) – конкретно документирано представяне на дадена структура;
- Двете понятия в голяма степен са взаимозаменяеми;
- Архитектурните структури се делят най-общо казано на 3 групи:
  - Модулни структури;
  - Структури на процесите;
  - Структури на разположението;

Модулни структури

- Елементите в модулните структури са модули – единици работа за изпълнение. Модулите предлагат поглед, ориентиран към реализацията на системата, без значение какво става по време на изпълнението;
- Някои въпроси, на които отговарят тези структури:
  - Коя функционалност в кой модул се реализира?
  - Кои други модули може да използва (и използва) дадения модул?
  - Как са свързани модулите по отношение на специализация и генерализация (наследяване);

Структури на процесите

- Елементите са компоненти, които се проявяват по време на изпълнението (т.е. основните изчислителни процеси) и средствата за комуникация между процесите.
- Някои въпроси, на които отговарят тези структури:
  - Кои са основните изчислителни процеси и как те си взаимодействат?
  - Кои са основните споделени ресурси?
  - Как се развиват данните в системата?
  - Кои части от системата могат да работят паралелно?
  - Как се променя структурата на системата докато тя работи?

Структури на разположението

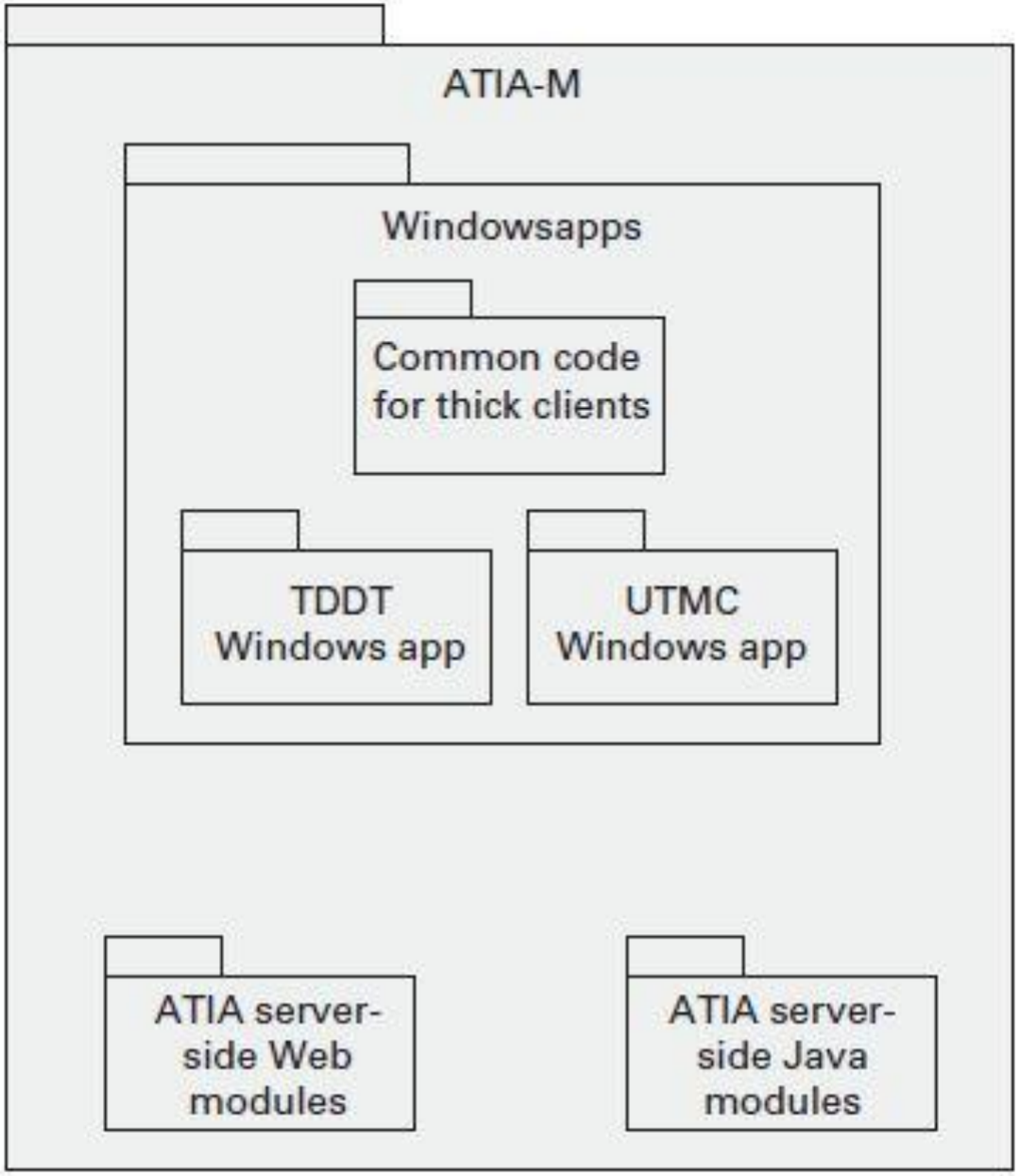
- Структурите на разположението показват връзката между софтуерните елементи и елементите на околната среда, в която се намира системата по време на разработката или по време на изпълнението;
- Някои въпроси, на които отговарят тези структури:
  - На кой процесор се изпълнява всеки от елементите?
  - В кои файлове се записва сорс кода на елементите по време на разработката?
  - Какво е разпределението на софтуерните елементи по екипи, които създават системата?

Модулни структури – Декомпозиция на модулите

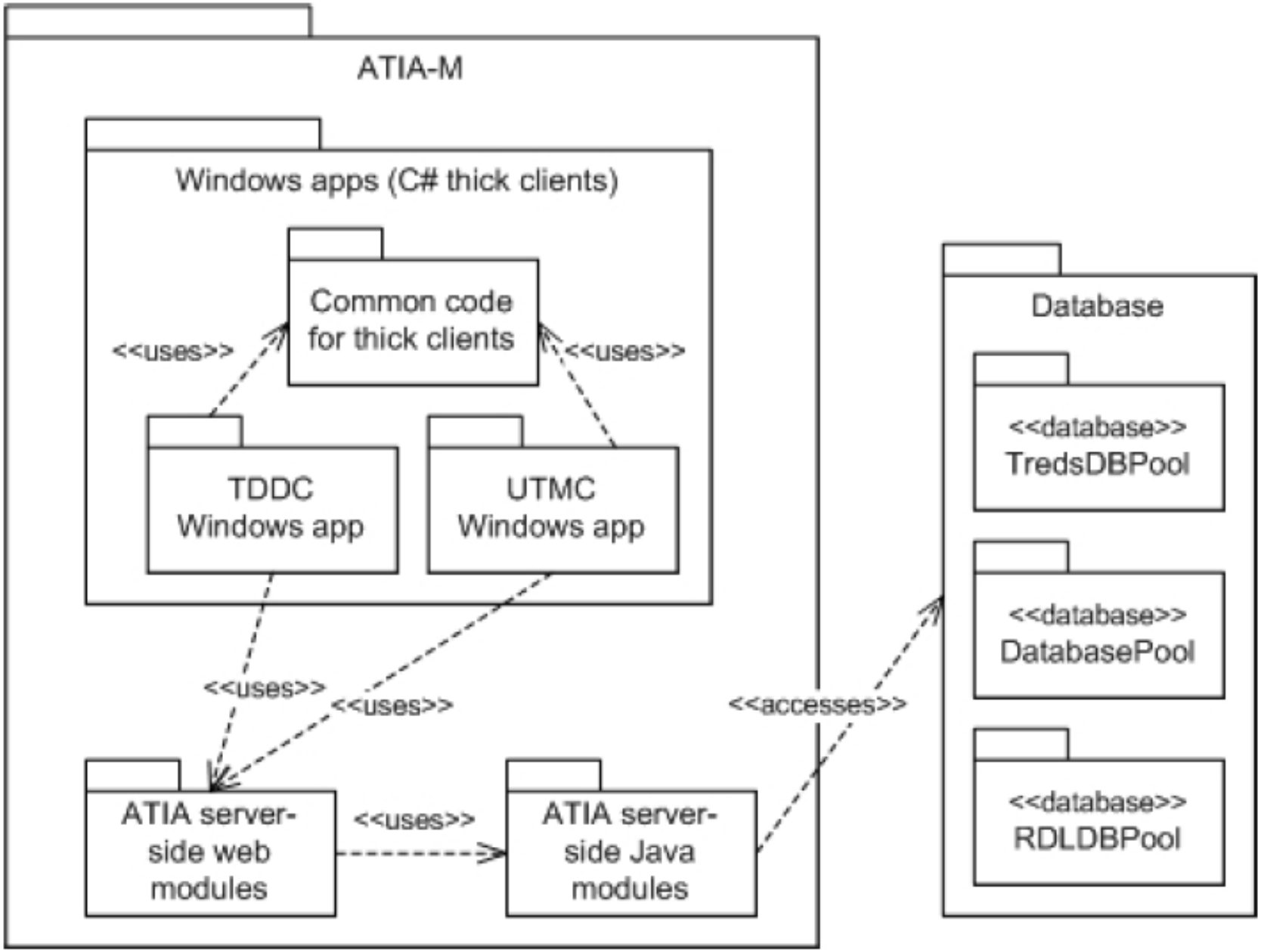
- Декомпозиция на модулите – връзките между модулите са от вида “X е под-модул на Y”;
- Това се прави рекурсивно до момента, в който елементите станат достатъчно прости, че да могат да бъдат разбрани лесно;
- Декомпозицията на модулите обуславя в голяма степен възможността за лесна промяна, като обособява логически свързани функционалности на едно място;
- Много често служи и като основа на разпределението на работата между екипите на изпълнителя;

Модулни структури – Употреба на модулите

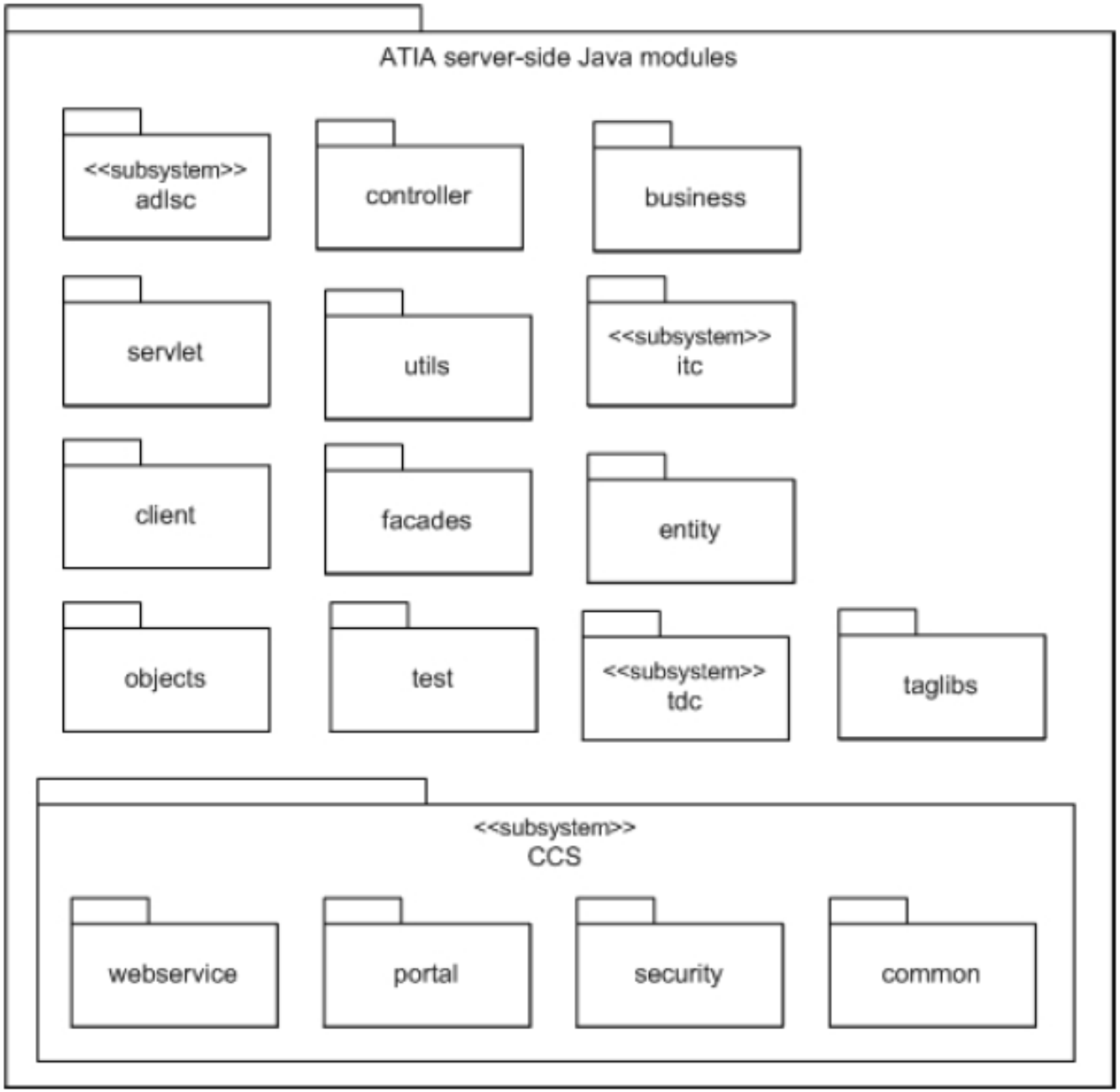
- Употреба на модулите – връзките между модулите са от вида “X използва Y”;
- Ако има нужда от по-детайлно описание, може връзките да са насочени към конкретен интерфейс или ресурс на модула;
- Структурата за употребата на модули обуславя възможността за лесно добавяне на нова функционалност, обособяване на [в голяма степен] самостоятелни подмножества от функционалност, както и позволява последователната разработка, много важна и мощна техника за работа;



Source: Paul Clements et all., Documenting Software Architectures: Views and Beyond, Pearson Education, 2011



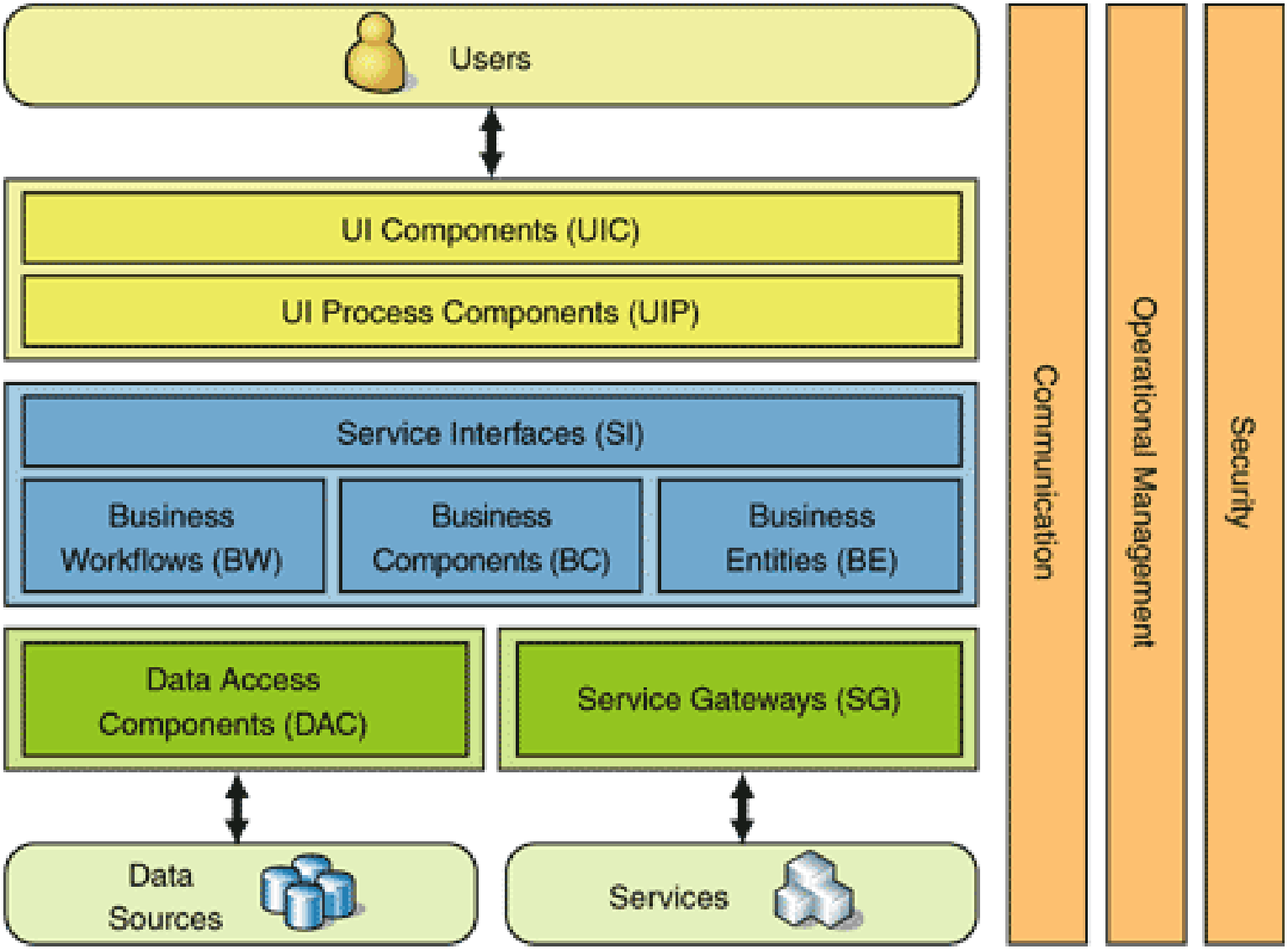
Source: Paul Clements et all., Documenting Software Architectures: Views and Beyond, Pearson Education, 2011



Source: Paul Clements et all., Documenting Software Architectures: Views and Beyond, Pearson Education, 2011

Модулни структури – Структура на слоевете

- Като частен случай на структурата на употребата на модули е структурата на слоевете – когато върху употребата са наложени стриктни правила се обособяват слоеве;
- Модулите от слой номер N могат да се възползват само от услугите на модулите от слой номер N-1;
- Слоевете често са реализирани като виртуални машини или обособени подсистеми, които скриват детайлите относно работата си от следващия слой;
- Не е прието (и е признак на лошо възпитание) слоеве да се прескачат;
- Структурата позволява без особени сътресения да бъде подменен цял един слой (напр. да се смени СУБД);

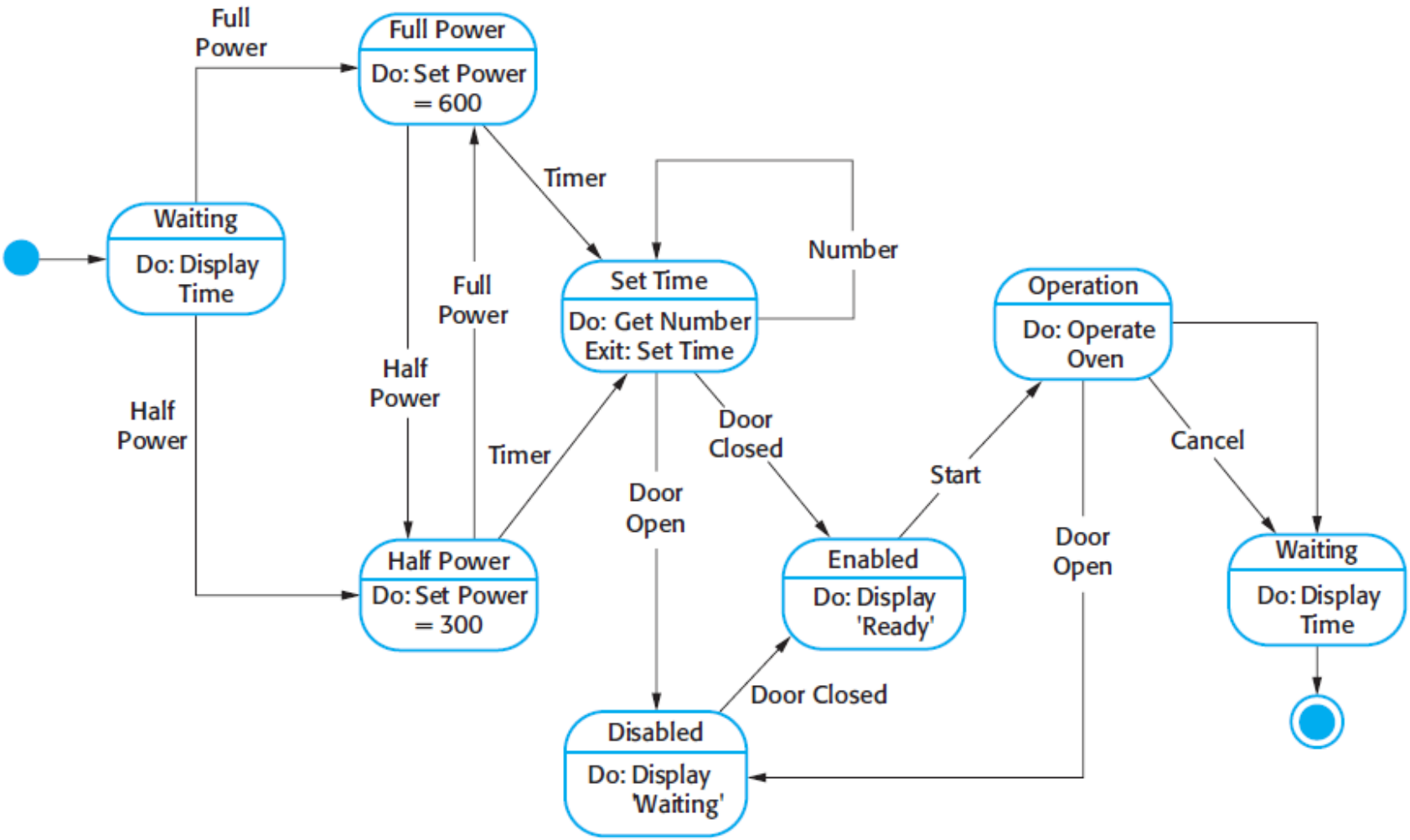

Users
UI Components (UIC)
UI Process Components (UIP)
Service Interfaces (SI)
Business Workflows (BW)
Business Components (BC)
Business Entities (BE)
Data Access Components (DAC)
Service Gateways (SG)
Data Sources
Services
Communication
Operational Management
Security

Модулни структури – Йерархия на класовете

- Йерархия на класовете – в терминологията на ООП, модулите се наричат “класове”, а в настоящата структура връзките между класовете са от вида “класът X наследява класа Y” и “обекта X е инстанция на клас Y”;
- Тази структура обосновава наследяването – защо подобни поведения или въобще функционалности са обособени в супер-класове или пък защо са дефинирани под-класове за обслужване на параметризирани различия;

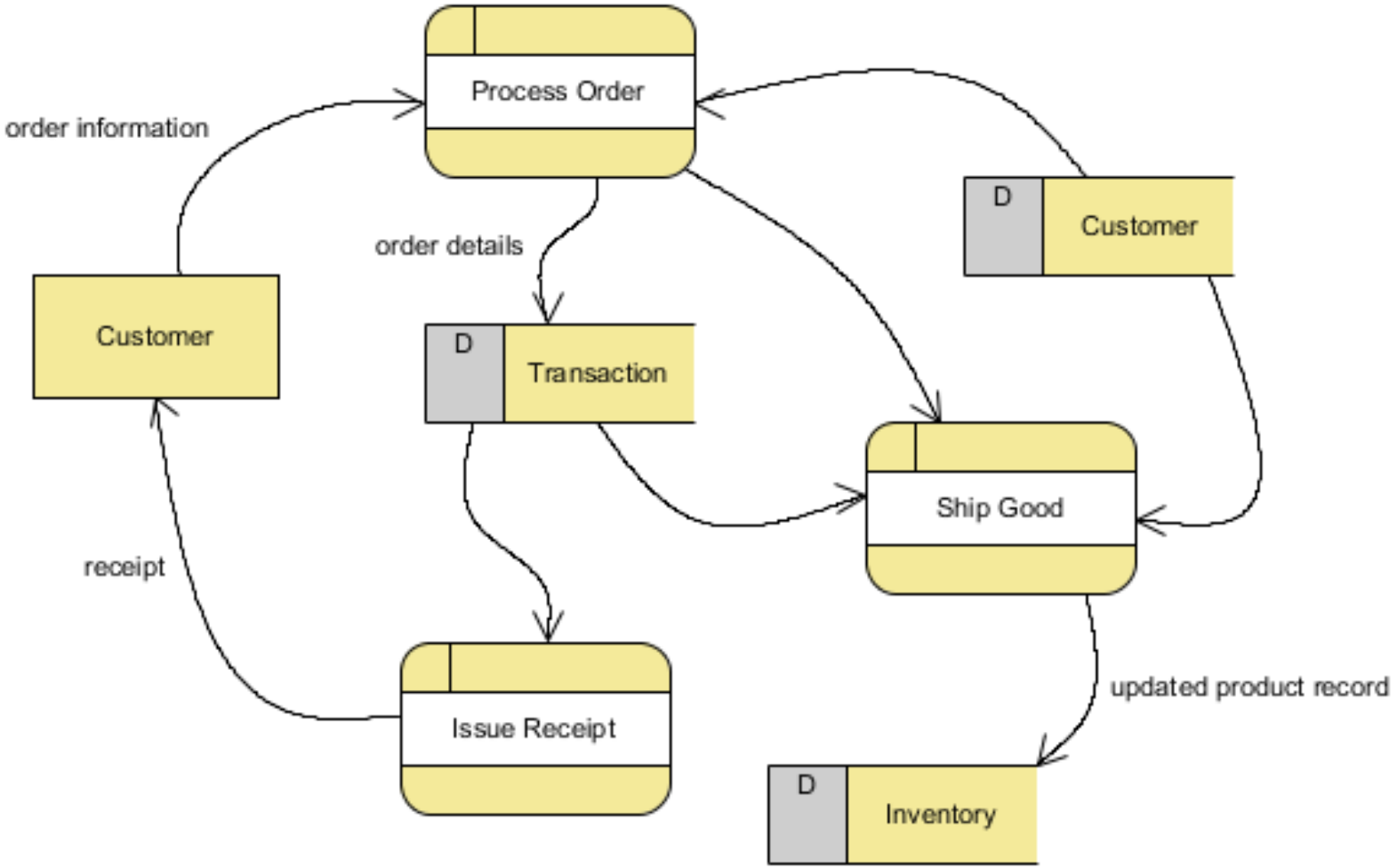
Структура на процесите

- Структура на процесите – елементите са процеси (или нишки), изпълнявани в системата (компоненти) и комуникационни, синхронизационни или блокиращи операции между тях (конектори); Връзките между тях (attachments) показват как компонентите и конекторите се отнасят помежду си;
- Структурата е полезна, тъй като има отношение по въпросите на бързодействието по време на изпълнението и високата надеждност.

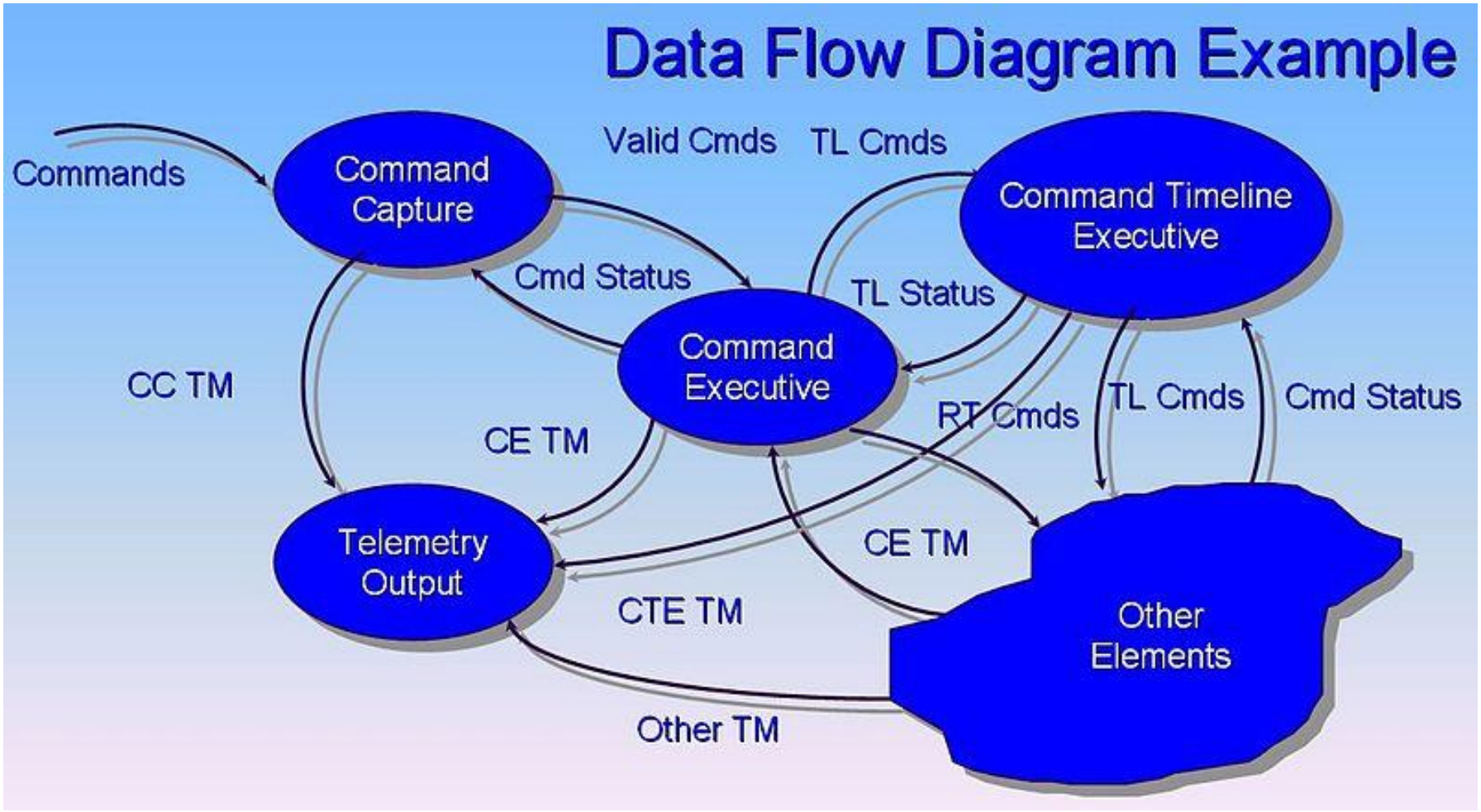


Source: Sommerville, I. (2016). Software Engineering. 10th edition. Pearson Education

Структура на потока на данните



Source: visual-paradigm.com

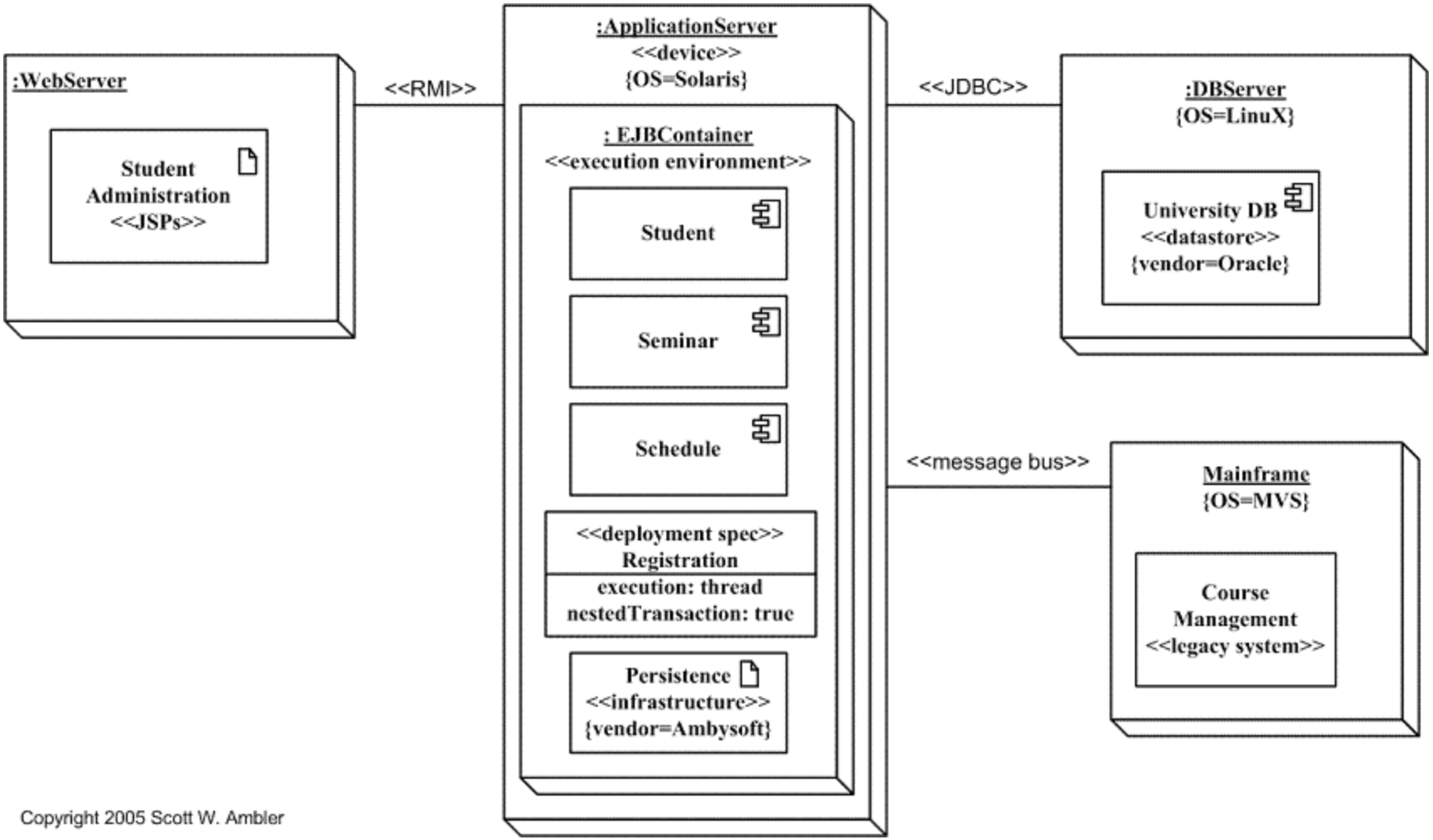


Source: wikipedia

Структури на разположението – Структура на внедряването

- Структура на внедряването – показва как софтуера се разполага върху хардуера и комуникационното оборудване;
- Елементите са процеси, хардуерни устройства и комуникационни канали;
- Връзките са напр. “внедрен върху” или “мигрира върху”;
- Представлява интерес при разпределени системи и позволява да се разберат особеностите относно бързодействието, интегритета на данните, надеждността, сигурността и т.н.;

Deployment structure



Структури на разположението – Файлова структура

- Файлова структура – показва кой модул къде се помещава, по време на различните фази на реализация;
- Структурата е критична за управлението на дейностите по разработка и за създаването и поддържането на обкръжение за build-ове;

Структури на разположението – Разпределение на работата

- Разпределение на работата – показва кой модул от кой екип се реализира;
- Елементите са модули и екипи, а връзките са кой модул от кой екип се разработва;
- Под “кой екип” не се има предвид конкретен списък от хора, а по-скоро виртуална група хора с подходящ опит, знания и умения;
- Архитектът трябва да знае какви хора са необходими за изработка на модулите и да участва във вземането на управленчески решения;
- Структурата помага и за това дадени общи функционалности да бъдат обособени и разработени от един екип, вместо всеки сам да си ги прави;

Кои структури да използваме?

- Зависи от системата;
- Зависи и от архитектурата
- Добра практика е да се използва поне по една структура от всяка от гореизброените групи структури

4 + 1 модел на софтуерната архитектура

1) Логически изглед – показва основните абстракции в системата, като обекти, класове и компоненти
2) Изглед на процесите – показва системата като съвкупност от взаимодействащи си процеси по време на изпълнение
3) Изглед на кода – Показва как отделните елементи на системата се разполагат във файлове код
4) Физически изглед – показва как софтуерните компоненти са разпределени между хардуерните възли в системата

+1) Съответните сценарии на употреба

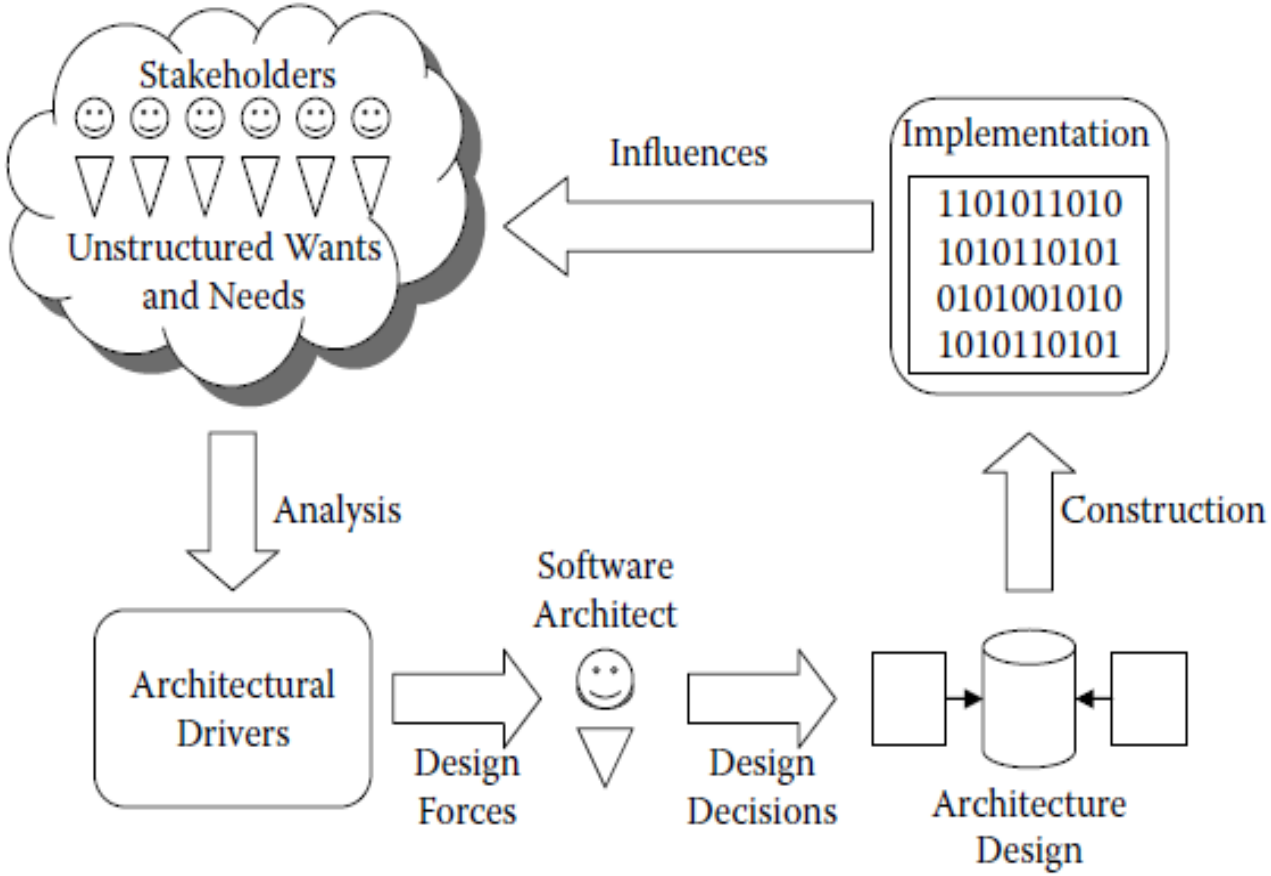
За и против софтуерната архитектура

- Няма софтуерна система без архитектура
- По важни са следните въпроси:
  - Мислена ли е архитектурата от гледна точка на удовлетворяване на някакви изисквания към системата
  - Доколко имплементацията (кодът на системата) се придържа към документацията.
  - Старее ли софтуера?
- Можете ли да дадете примери за добри или лоши архитектури?

От какво се определя СА?

- Широко разпространено е схващането, че СА зависи само от изискванията;
- Истината е, че се намесват и много други фактори на обкръжението (environment), а именно:
  - технически, бизнес и социални влияния;
  - опит, знания и умения на архитекта;
  - съвременните технологии;
- От друга страна, самото създаване на СА повлиява върху обкръжението, т.е. процесът е цикличен;

Цикличен процес на създаване на архитектурата



Anthony J. Lattanze, Architecting Software Intensive Systems: A Practitioner's Guide, 2009 Taylor & Francis Group, LLC

Понятие за stakeholder

- Stakeholder – Заинтересовано Лице (ЗЛ)
  - Това са всички, които имат отношение към създаването на софтуерната система – напр. собствениците, управителите, специалистите по продажби, ръководителя на проекта, разработчиците, екипа по поддръжка, различни прослойки от страна на клиента, крайните потребители и т.н.

Понятие за stakeholder

- Всички те имат разнопосочни интереси, напр.:
  - Да се държи по определен начин;
  - Да работи добре на определен хардуер;
  - Да може лесно да се променя;
  - Да се стане бързо;
  - Да стане евтино;
  - Да я правят хора с конкретни умения;
  - Да е многофункционална;
  - и т.н.

Влияние на ЗЛ върху архитектурата

- Тези интереси най-често си противоречат;
- Архитектът е в неблагоприятна позиция – какъвто и ход да предприеме, все някой от списъка със ЗЛ ще е недоволен;
- Ролята му е да балансира между различните ЗЛ за бъдат конкретните интереси отразени в спецификацията на изискванията!

Влияние на организацията върху архитектурата

- Основното влияние идва от целите, заради които се създава системата (изискванията ги отразяват най-пълно);
- Други влияния са:
  - Текущо състояние на организацията;
  - Употреба на предишни разработки;
  - Организационна структура;
  - Стратегия за дългосрочни инвестиции;

Влияние на технологиите

- Частен случай на влиянието на опита и средата на архитекта е влиянието на текущите технологии:
  - Индустриални стандарти;
  - Най-добри практики;
  - Преобладаващи инженерни техники;
- В настоящия момент модерни са уеб-базираните, ориентирани към услуги софтуерни архитектури.

Влияние на опита на архитекта

- Знанията и уменията на архитекта влияят върху създаваната СА:
  - Ако архитектът има положителен опит с даден подход, вероятно ще го използва отново;
  - Обратно, ако резултатите са били катастрофални, най-вероятно ще се въздържи;
  - Подходът ще зависи и от това къде, какво и колко е учил и чел архитекта;
  - Дали се е сблъсквал с успешни/неуспешни подходи и/или реализации;
  - Наклонности за експерименти;

Реалната картина

- В много редки случаи изискванията, породени от бизнес целите, както и различните влияния, са напълно разбрани, обяснени и документирани
- Това води до конфликти между различните ЗЛ, които трябва да се разрешават
- За целта архитекта трябва да разбере същността, източниците и приоритетите на различните ограничения и трябва да управлява нуждите и очакванията на ЗЛ
- Крайната цел е ЗЛ да (бъдат притиснати да) приближат позициите си така, че да се намери пресечна точка между противоречивите на пръв поглед изисквания

Важни за архитекта качества

- Казаното до тук предполага, че за да бъде един архитект успешен, той се нуждае от качества като:
  - Отлично познаване на технологиите
  - Отлично аналитично и абстрактно мислене
  - Комуникативност, дипломатичност и умение за убеждаване и въобще за водене на преговори

Важни за софтуерния архитект дейности

- Архитектът взема участие в следните дейности:
  - Вземане на бизнес решения за създаване на системите;
  - Разбиране на изискванията;
  - Създаване или избор на архитектура;
  - Документиране на СА;
  - Анализ и оценка на СА;
  - Създаване на системата;
  - Следене за наличие на съответствие между системата и СА;

Вземане на бизнес решения

- Освен извършването на маркетингово проучване, за да се вземе решение за създаване на дадена система следва да се отговори на въпросите:
  - Каква е целевата функция?
  - Колко ще струва?
  - За колко време?
  - В каква среда ще работи (интерфейси с други системи)?
  - Някакви ограничения?
- Все въпроси, по които архитекта следва да вземе отношение. Ако той не участва във вземането на бизнес решението, вероятността за провал се увеличава.

Разбиране на изискванията

- Изискванията (функционални и нефункционални) обуславят СА;
- Те трябва да се дефинират по възможно най-недвусмислен начин;
- Ако архитектът участва в дефиницията на изискванията, вероятността да се създаде система, която отговаря на поставените бизнес цели е по-голяма.

Създаване или избор на архитектура

- Създаването или изборът на архитектура е същинската работа на архитекта;
- Съществената тук е, че успешен проект и разработка могат да се изградят само ако е налице идейна цялост, а идейна цялост може да се постигне само посредством последователен и подреден ***мисловен*** процес от страна на специализирани (малко на брой) архитекти;

Документиране на СА

- Втората част от същинската работа на архитекта;
- И най-добрата архитектура е безполезна, ако тя не бъде по подходящ начин представена на всички ЗЛ;
- Нюансът тук е, че формата, под която следва да бъде представена СА зависи от конкретните ЗЛ;

Анализ и оценка на СА

- Както при всеки процес на проектиране, и при създаване на СА най-вероятно има няколко варианта, които следва да се оценят и анализират и да се избере най-добрия;
- Архитектурите подлежат на оценка както по отношение на изпълняване на изискванията, така и по отношение на финансови параметри;

Създаване на системата

- Ролята на архитекта по време на създаването (implementation) на системата е основно да следни дали се спазват предписанията на СА;
- Това, че има прекрасна, добре документирана и преразказана архитектура е добре, но ако хората, които правят системата не я следват, ефектът е нулев.

Следене за съответствие

- След като системата бъда разработена и премине във фаза на поддръжка, архитектът трябва да следи за съответствието между СА и системата – по време на поддръжката се налагат промени; тяхната реализация следва да е съгласно принципите на архитектурата; от своя страна, СА също трябва да се адаптира към промените

Полза от софтуерната архитектура

- За улесняване на дискусиите и анализа на системите между заинтересованите лица
- За улесняване на разработката и поддръжката (в рамките на екипа/организацията)
- За създаване на формални модели на софтуерната система
  - Верификация/Валидация чрез симулация
  - Генерация на код