



Лекция 2

Основни концепции, описващи софтуерната разработка

DAAD Project
“Joint Course on Software Engineering”

Humboldt University Berlin, University of Novi Sad, University of Plovdiv,
University of Skopje, University of Belgrade, University of Niš, University of Kragujevac

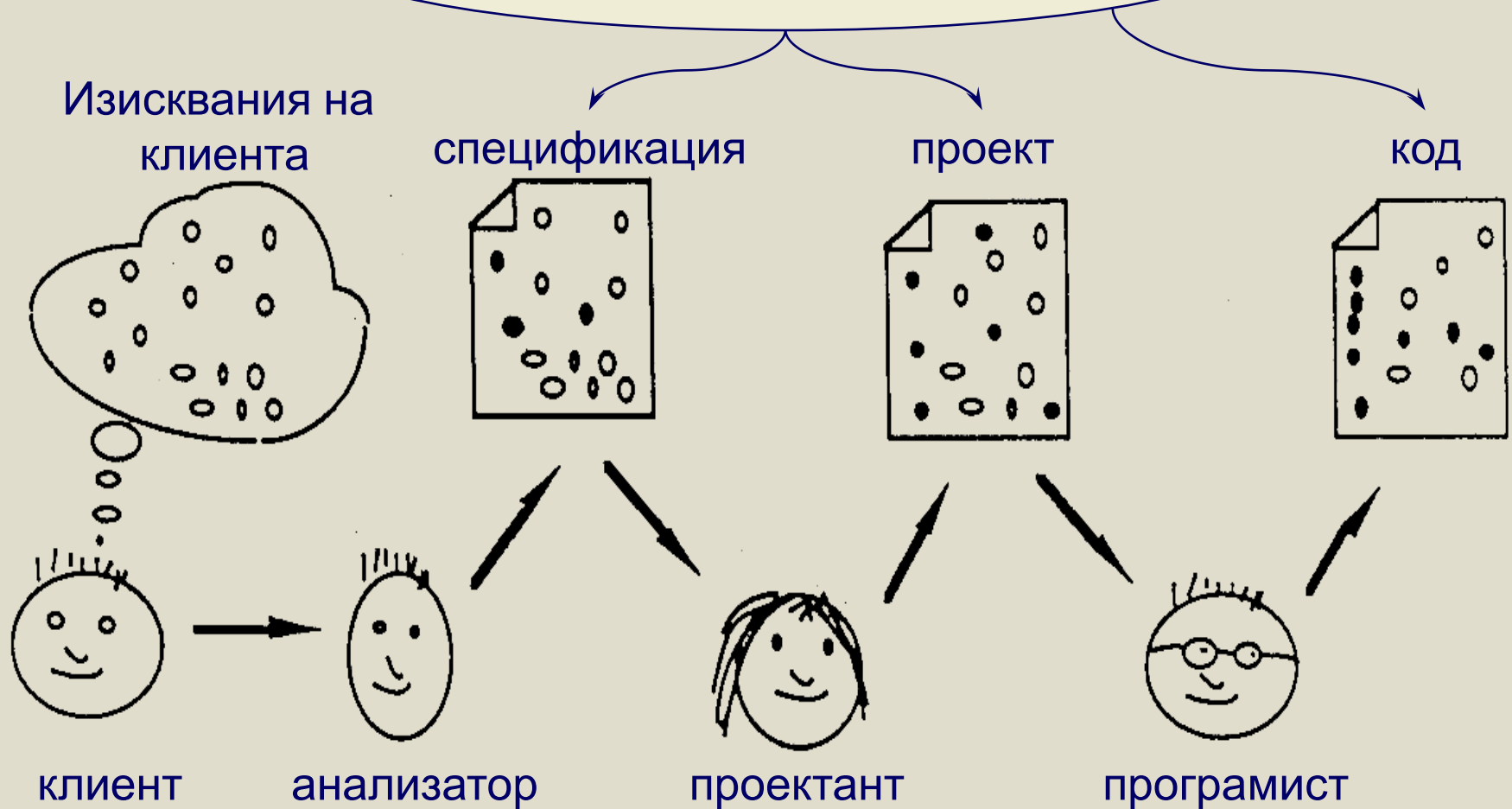
Parts of this topic use material from the textbook
H. Balzert, “Software-Technik”, Vol. 1, 2nd ed., Spektrum Akademischer Verlag, 2001

4. Основни концепции описващи софтуерната разработка

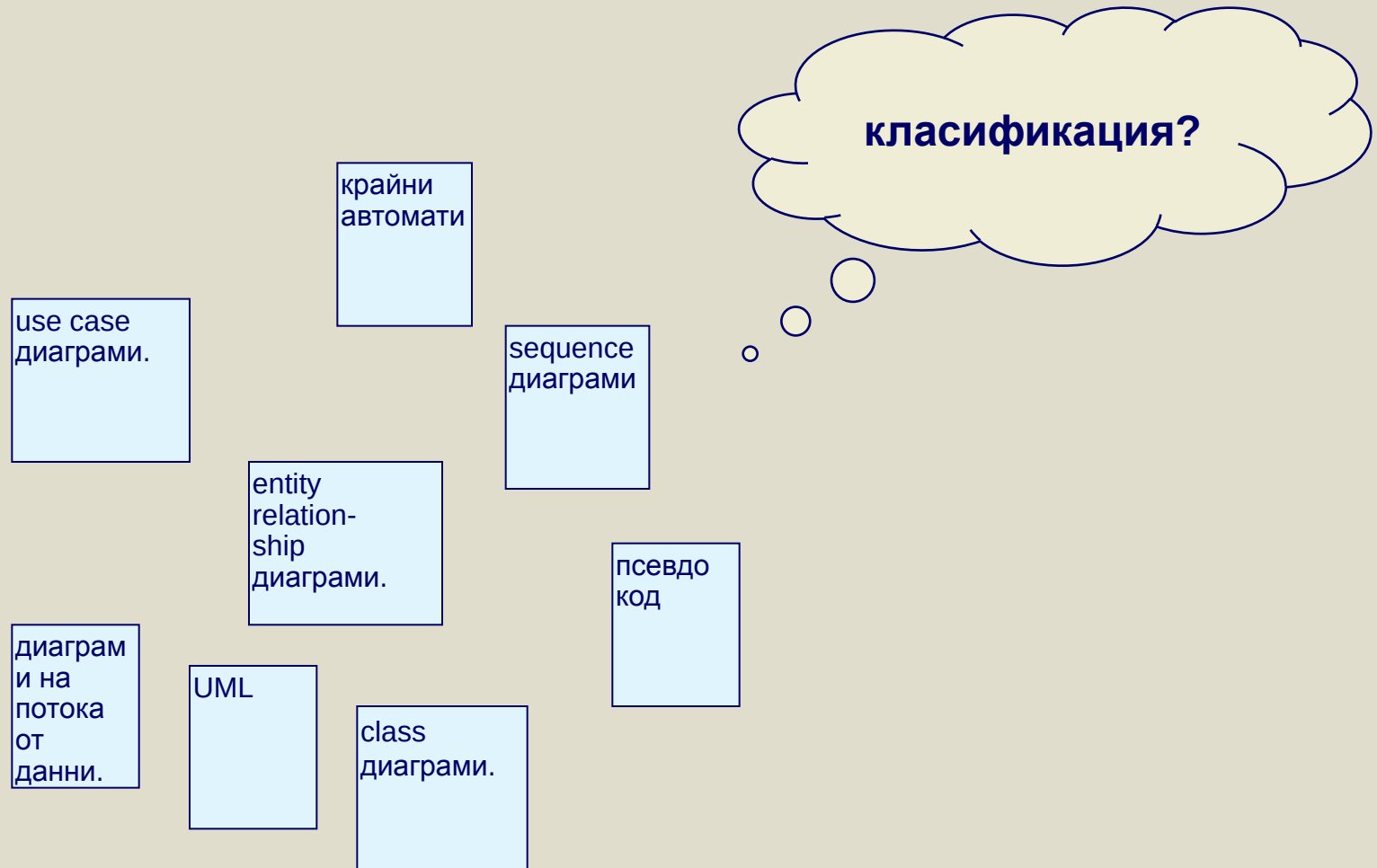
- a) Описание на резултатите от фазите на разработка на софтуер – документи при разработката на софтуер
- b) Дефиниции на основните концепции за описание на разработката на софтуер и тяхната класификация
- c) Основните концепции и фазите на разработка на софтуер
- d) Класификация на основните концепции според техните нотации
- e) Основните концепции и техните области на приложение
- f) Основните концепции и тяхното приложение в методите за анализ

Техники за описание на софтуерните ДОКУМЕНТИ

Проблем: Как да се опишат ?



Описание на софтуерните документи чрез ...



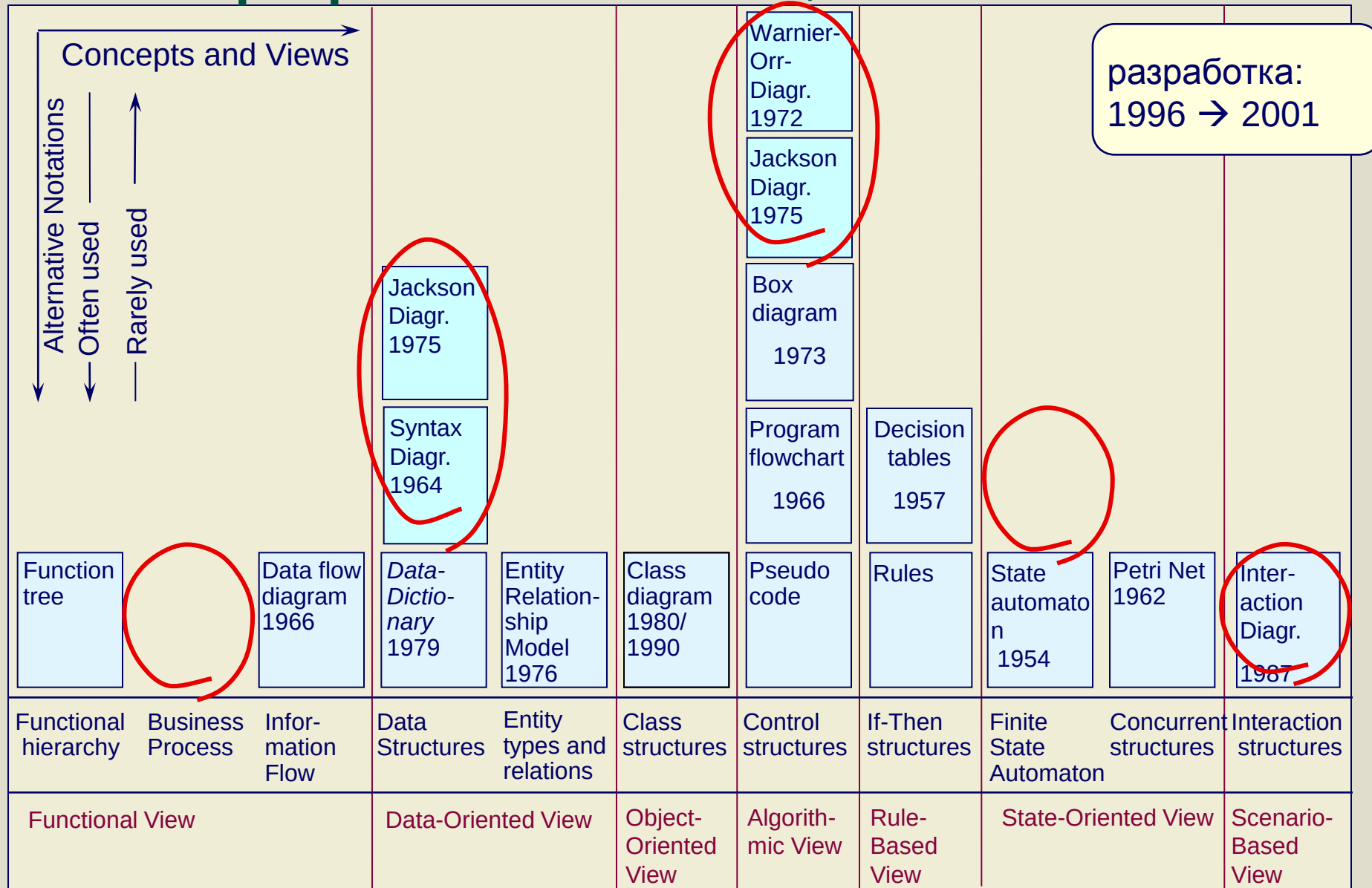
Основните концепции на ... софтуерната разработка Balzert vol. 1, 2nd edition 2001

Concepts and Views Alternative Notations Often used Rarely used												
Function tree	Use Case Diagram 1987	Data flow diagram 1966	<i>Data-Dictionary</i> 1979	Entity Relationship Model 1976	Class diagram 1980/1990	Pseudo code	Box diagram 1973	Program flowchart 1966	Decision tables 1957	Activity diagram 1997	Collaboration diagram	Sequence diagram 1987
Functional hierarchy	Business Process	Information Flow	Data Structures	Entity types and relations	Class structures	Control structures	If-Then structures	Finite State Automaton	Concurrent structures	Interaction structures		
Functional View			Data-Oriented View		Object-Oriented View	Algorithmic View	Rule-Based View	State-Oriented View		Scenario-Based View		

Основни концепции: основен подход



Основни концепции на ... софтуерната разработка Balzert vol. 1, 1st edition 1996



Softwareentwicklung bleibt Methodenmix

Softwareentwicklung als Wissenschaft und Softwareentwicklung als praktische Tätigkeit in Unternehmen haben etwas gemeinsam: die Heterogenität. Die technisch beeinflussten Meinungen über die richtige Softwaremethode sind gespalten – selbst im engeren Bereich der objektorientierten Methoden gibt es mehr als 50 Ansätze –, und der Reifegrad der Softwareentwicklung in Firmen reicht von „kein Prozeß“ bis zu „perfekt durchorganisiert“. Dabei wäre es sicher besser, irgendeine der existierenden Entwicklungsmethoden zu verwenden als überhaupt keine. Eine stabile Größe von Anbeginn der problembewußten Softwareentwicklung ist die zugehörige Diskussion: Softwareentwicklung – Kunst oder Ingenieurwissenschaft?

Der hier zugrunde liegende Gegensatz ist Geist (Improvisation und knifflige Algorithmen) und Disziplin (Prozesse). Für den Bau großer Systeme ist die Notwendigkeit von Disziplin unumstritten.

Aber die Softwaretechnik hat sich auch als Wissenschaft profiliert und in den letzten Jahren wichtige Schritte gebracht: verteilte Objekte, Netzwerke, Betriebssysteme wie Java. Die Softwaretechnik ist zu einem der innovativsten Industriezweige geworden. Schlüsseltechnologie für Innovation selbst ist damit ein zu balancierendes Tripel: Softwaretechnik, Innovation.

Den Administrator sprechen vor allem die Softwaretechnik und Innovation. Manager und Politiker müssen sie zusammenbringen.

Sicher ist auch die Einführung neuer Produkte mühsam, aber das Problembewußtsein ist größer und deshalb heute wichtiger sind die aktiven und konstruktiven Faktoren der neuen Softwaretechnik und der Softwareentwicklung, aber hier liegt möglicherweise der dringend benötigte Antrieb für die Softwareentwicklung und Industrie. Traditionell ist die Softwareentwicklung in Deutschland wohl geregelter. Der Schwerpunkt zukünftiger Softwareentwicklung in Deutschland sollte die Softwareentwicklung sein. Die Softwareentwicklung liegt im Zentrum der neuen Technik kommen sowohl die Softwareentwicklung als auch die Softwareentwicklung als auch die Softwareentwicklung.

Softwareentwicklung als Wissenschaft und Softwareentwicklung als praktische Tätigkeit in Unternehmen haben etwas gemeinsam: die Heterogenität. Die technisch beeinflussten Meinungen über die richtige Softwaremethode sind gespalten – selbst im engeren Bereich der objektorientierten Methoden gibt es mehr als 50 Ansätze –, und der Reifegrad der Softwareentwicklung in Firmen reicht von „kein Prozeß“ bis zu „perfekt durchorganisiert“. Dabei wäre es sicher besser, irgendeine der existierenden Entwicklungsmethoden zu verwenden als überhaupt keine. Eine stabile Größe von Anbeginn der problembewußten Softwareentwicklung ist die zugehörige Diskussion: Softwareentwicklung – Kunst oder Ingenieurwissenschaft?

Softwareentwicklung bleibt Methodenmix

Softwareentwicklung als Wissenschaft und Softwareentwicklung als praktische Tätigkeit in Unternehmen haben etwas gemeinsam: die Heterogenität. Die technisch beeinflussten Meinungen über die richtige Softwaremethode sind gespalten – selbst im

Софтуерната разработка е смесица от методи

Софтуерната разработка като наука и като практическа дейност има нещо общо : хетерогенност.

Kunst oder Ingenieurwissenschaft?

- Разнообразие от подходи в основните концепции:
- различни изгледи на софтуерния продукт трябва да бъдат възможни
- (различни аспекти на софтуерния продукт)

Основни концепции: принципи

- ▶ **независими от фазите**

- използваеми в различни фази

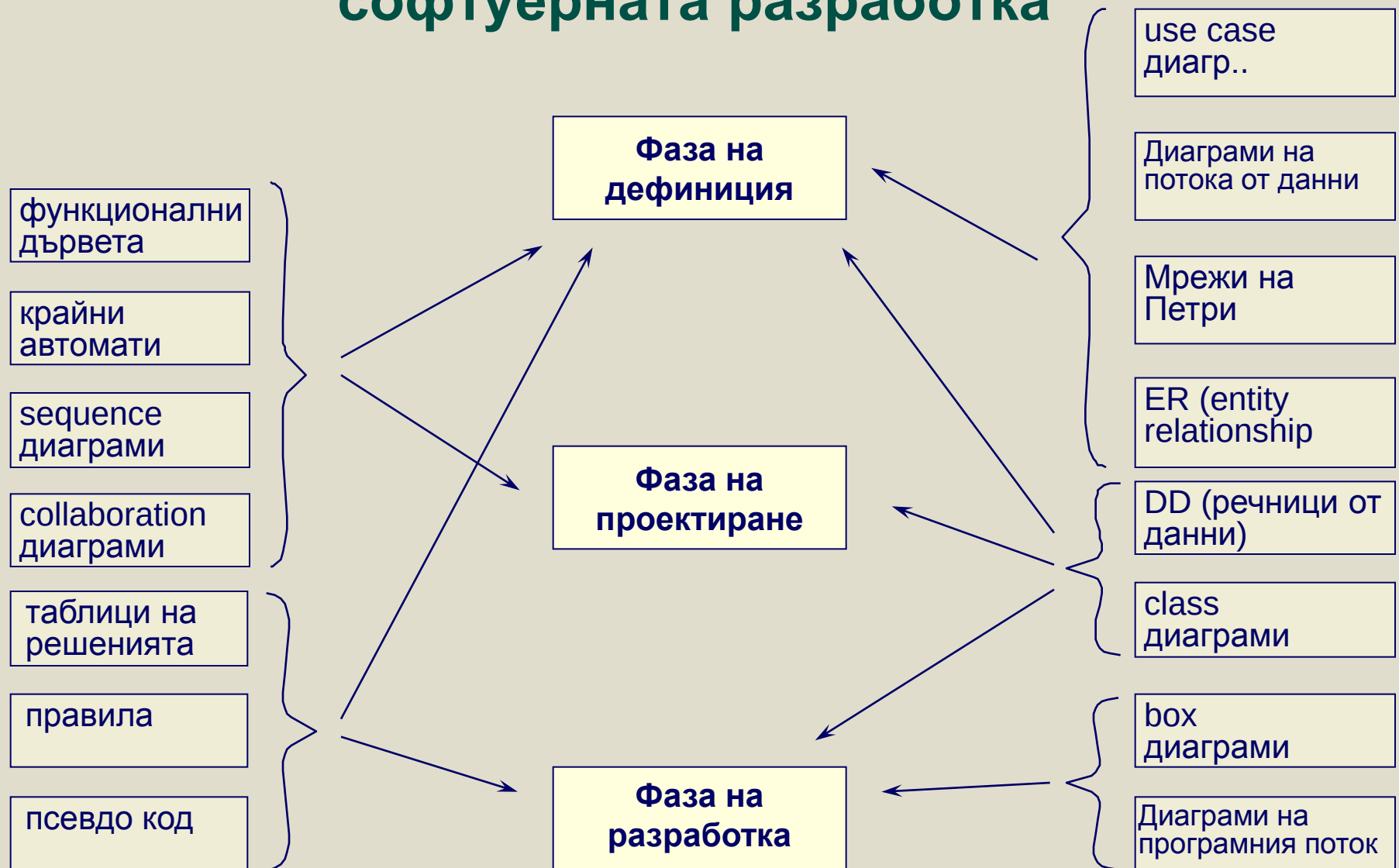
- ▶ **независими от приложението**

- типа на приложението определя само избора на подходяща основна концепция

- ▶ **Независими от метода на развой**

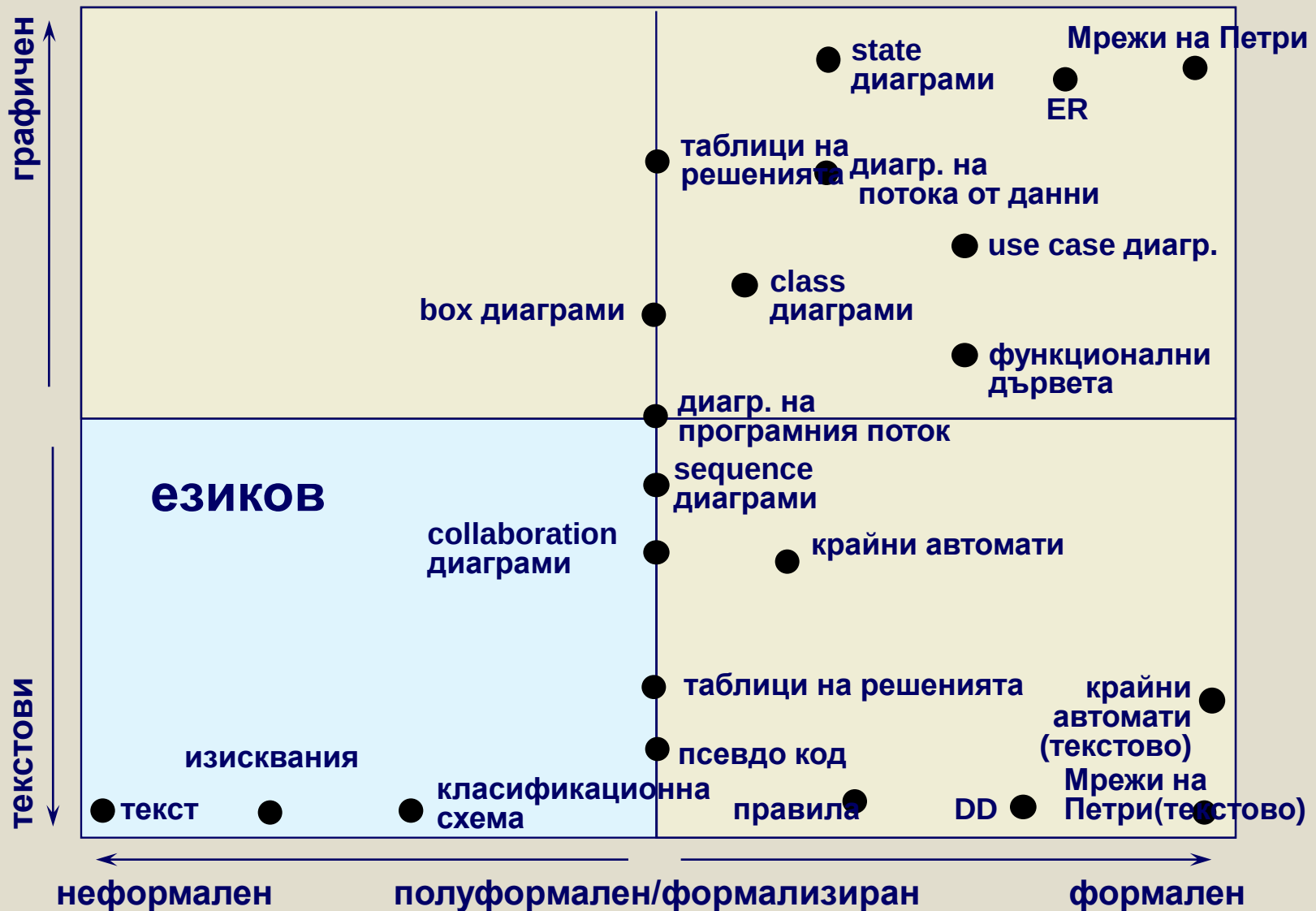
- Основните концепции могат да се комбинират за общи методи (например: структурен анализ, ООА с UML ...)

Основните концепции и фазите на софтуерната разработка



Легенда: A → B: A е използвано в B

Класификация на основните концепции според техните нотации



Основните концепции и приложни области

приложни области	сложност...	основни концепции
↑ ↓ технически-научен ↑ ↓ реално време ↑ ↓ взаимодействие човек-компютър	...на данни	▶ Речници от данни ▶ ER (Entity Relationship) ▶ (крайни автомати)
	...на функции	▶ class диаграми ▶ use case диаграми ▶ диаграми на потока от данни ▶ функционални дървета
	...на алгоритми	▶ псевдо код ▶ диаграми на потока на програмата ▶ box диаграми ▶ таблици на решенията ▶ правила
	...на системната среда	▶ <u>Диаграми на потока от данни</u>
	...на зависещо от време поведение	▶ Мрежи на Петри ▶ Крайни автомати ▶ activity диаграми ▶ sequence диаграми ▶ collaboration диаграми
	...на потребителски интерфейс	▶ <u>графична спецификация</u> ▶ (Мрежи на Петри) ▶ (крайни автомати) ▶ Контролни структури ▶ правила
↑ ↓ легенда: () = ограничено		

Основните концепции на ... софтуерната разработка:

допълнения

Balzert vol. 1, 2nd edition 2001

Algebraic Specification 1972	Hoare logic 1969	Z	Structure Chart	Grammars	EBNF	Package Diagram	Component Diagram	State Chart 1987	SDL
------------------------------	------------------	---	-----------------	----------	------	-----------------	-------------------	------------------	-----

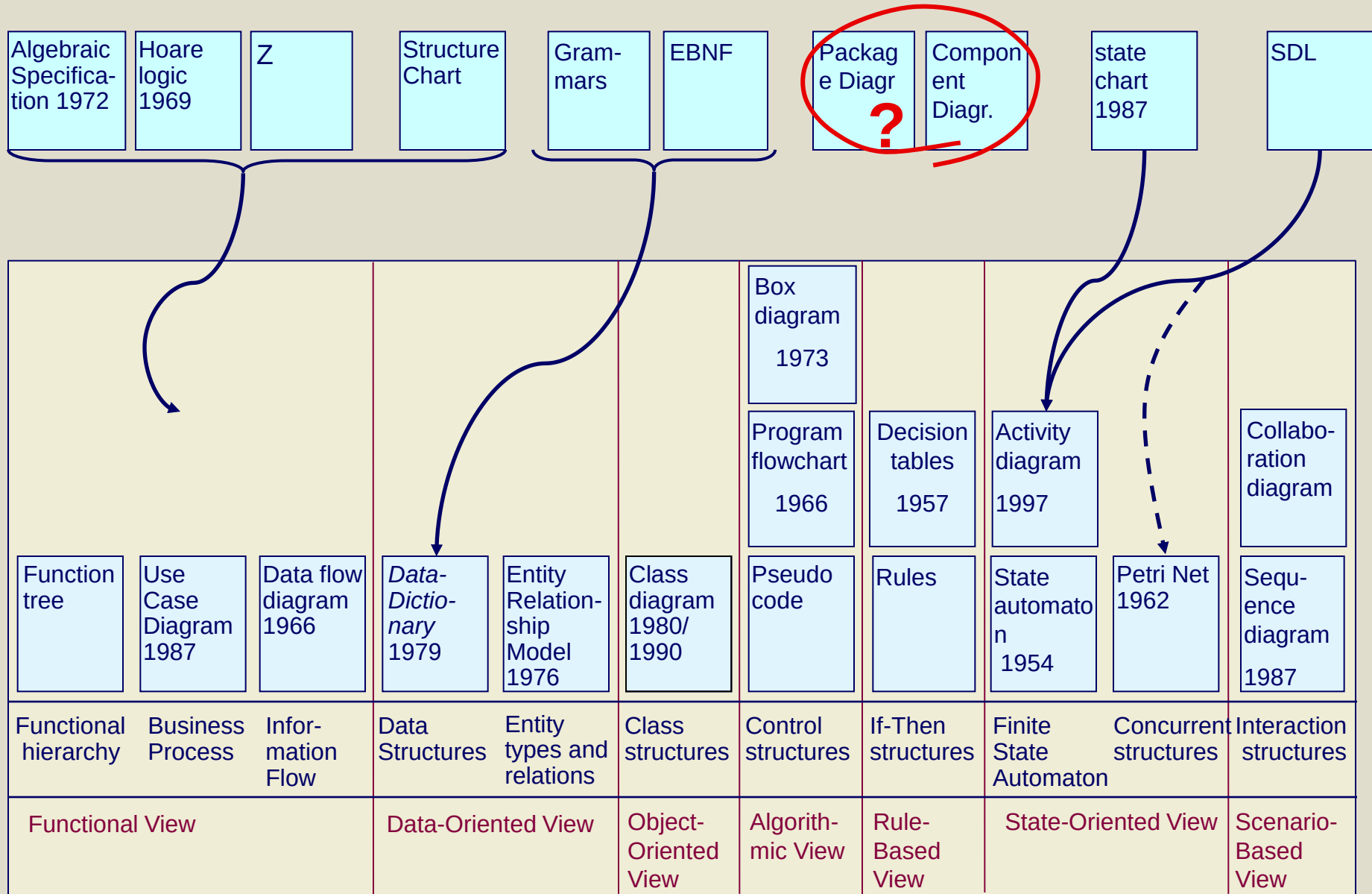
Основни концепции:

- пълни?
- допълнителни форми на нотации?

класификация на 3 нива
Гъвкава схема

Основни концепции: • пълни? • допълнителни форми на нотации?						Box diagram 1973	Гъвкава схема				
						Program flowchart 1966	Decision tables 1957	Activity diagram 1997		Collaboration diagram	
Function tree	Use Case Diagram 1987	Data flow diagram 1966	Data-Dictionary 1979	Entity Relationship Model 1976	Class diagram 1980/1990	Pseudo code	Rules	State automaton 1954	Petri Net 1962	Sequence diagram 1987	
Functional hierarchy	Business Process	Information Flow	Data Structures	Entity types and relations	Class structures	Control structures	If-Then structures	Finite State Automaton	Concurrent structures	Interaction structures	
Functional View			Data-Oriented View		Object-Oriented View	Algorithmic View	Rule-Based View	State-Oriented View		Scenario-Based View	

Основни концепции: класификация на допълненията



Основни концепции: детайлна класификация на допълненията

Разширения на изгледите ...

Разширения на аспектите ...

Z

Structure Chart

Grammars

EBNF

Data-Dictionary
1979

ER Entity Relationship
1976

Class diagram
1980/1990

Package Diagram

Component diagram

Function tree

Use Case Diagram
1987

Data flow diagram
1966

Hoare logic

Algebraic Specification
1972

source code subsystems

binary code subsystems

Functional hierarchy

Business Process

Information Flow

effect of a function

interaction between functions

effect of functions to data

Data structures

Entity types and relations

Class structures

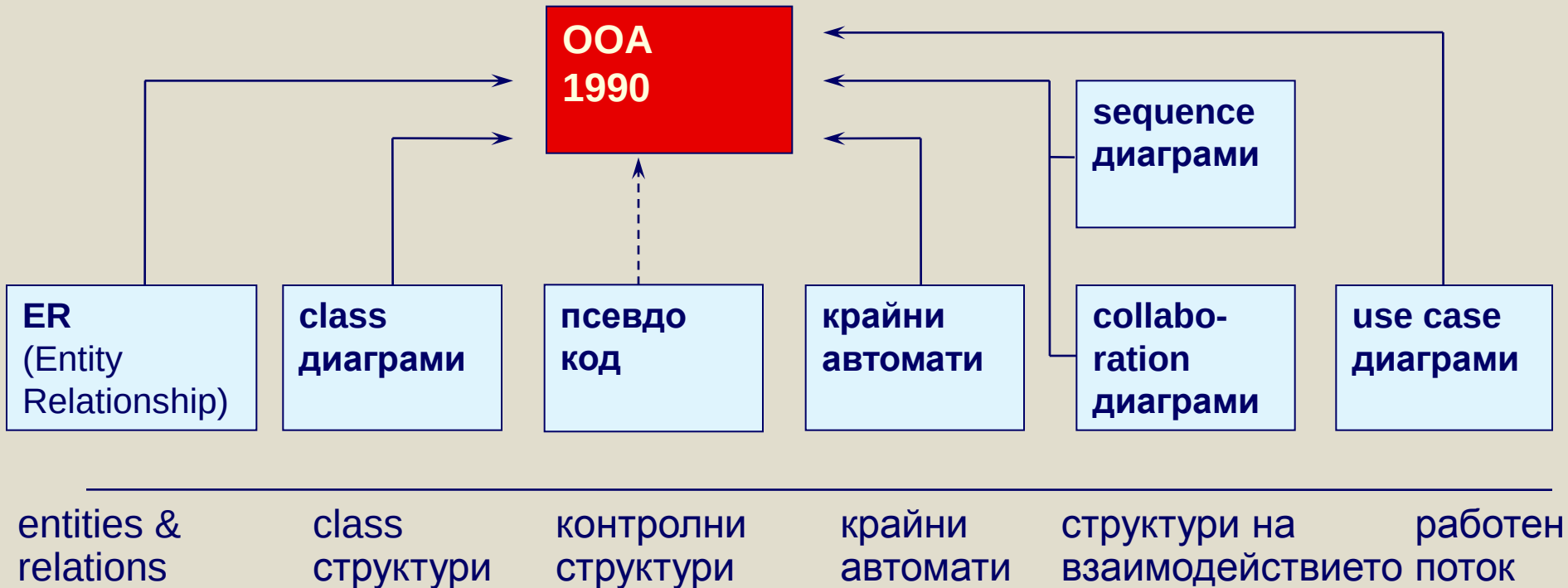
component-oriented view

Functional View

Data-Oriented View

Object-Oriented View

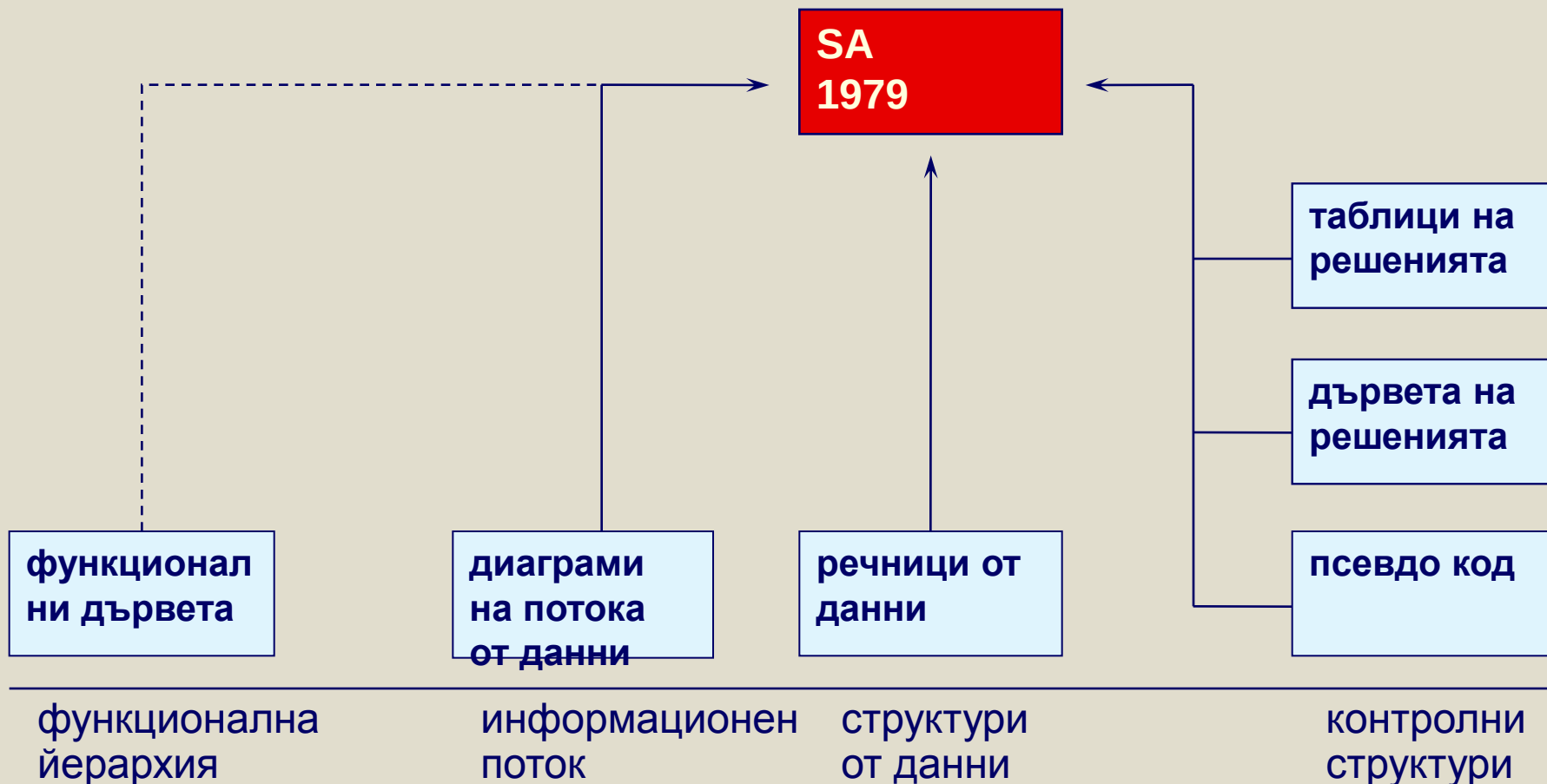
Комбинация от основни концепции в обектно-ориентирания анализ



легенда: A → B: A се съдържа в B

A ---> B: A напълно се съдържа в B

Комбинация от основни концепции в структурния анализ



легенда: A → B: A се съдържа в B
A ----> B: A напълно се съдържа в B

Комбинация от основни концепции в структурния анализ и real-time анализ

