TEHNIČKO VELEUČILIŠTE U ZAGREBU
POLYTECHNICUM ZAGRABIENSE

Objektno orijentirani razvoj programa

ISVU: 130938/19970

Dr. sc. Danko Ivošević, dipl. ing.
Predavač

Akademska godina 2021./2022.
Ljetni semestar

OBJEKTNO ORIJENTIRANI RAZVOJ PROGRAMA

UNIFIED MODELING LANGUAGE (UML)

Unified Modeling Language

- UML je jezik za:
  - vizualizaciju;
  - specifikaciju;
  - oblikovanje i
  - dokumentiranje artefakata programske potpore.
- Omogućava različite poglede na model
- ‘de facto’ standardni jezik programskog oblikovanja
- UML je posebice prikladan za specificiranje *objektno usmjerene* arhitekture programske potpore.
- Dijelovi UML-a pogodni su u specificiranju i drugih arhitektura.
- Omogućava:
  - Višestruke međusobno povezane poglede
  - Poluformalnu semantiku izraženu kao meta-model
  - Jezik za opis formalnih logičkih ograničenja

Temelji UML-a

- **UML** je jezik za:
  - Vizualizaciju
  - Specifikaciju
  - Oblikovanje (i konstruiranje)
  - Dokumentiranje artefakata programske potpore.
- UML je posebice prikladan za specificiranje **objektno usmjerene arhitekture programske potpore**.
- Dijelovi UML-a pogodni su u specificiranju i drugih arhitektura.

- **Arhitekture programske potpore je struktura ili strukture sustava koji sadrži elemente, njihova izvana vidljiva obilježja i odnose između njih.**

Prednosti UML-a

- **Otvoren standard.**
- **Podupire cijeli životni ciklus oblikovanja programske potpore.**
- **Podupire različite domene primjene.**
- **Temeljen je na iskustvu i potrebama zajednice oblikovatelja i korisnika programske potpore.**
- **Podržavaju ga mnogi alati.**

Mnogo dionika, mnogo pogleda

- **Arhitekturu programske potpore različito vidi:**
  - **Korisnik-kupac**
  - **Projekt manager**
  - **Inženjer sustava**
  - **Osoba koje razvija sustav**
  - **Arhitekt**
  - **Osoba koja održava sustav (engl. *maintainer*)**
  - **Drugi dionici**

- ***Višedimenzionalna stvarnost***
- **Mnogo dionika rezultira u**
  - **višestrukim pogledima, višestrukim nacrtima**

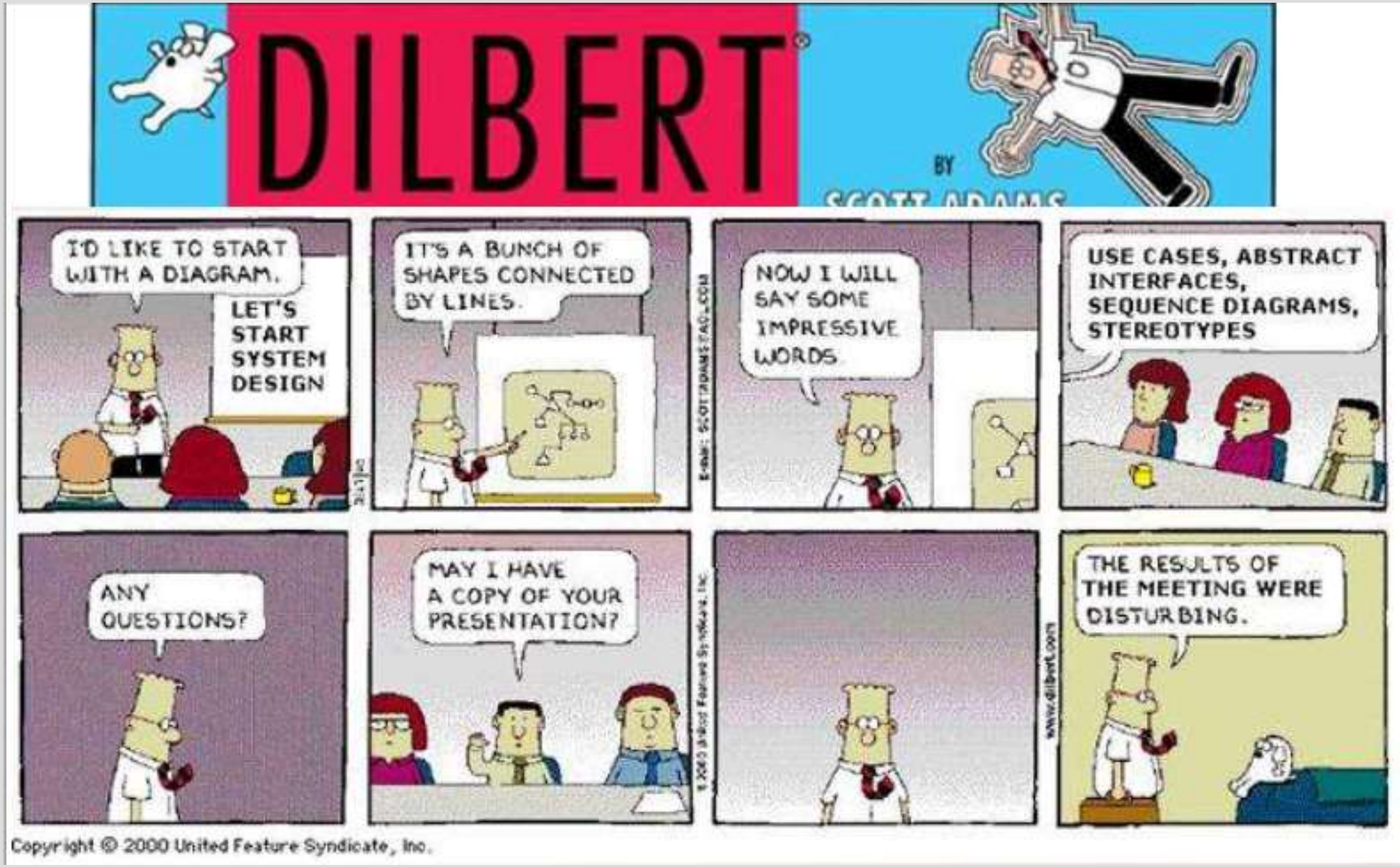
Koliko pogleda ?

- **Pogledi odgovaraju nekom kontekstu.**
- **Svi sustavi ne trebaju sve poglede:**
  - **Jedan procesor, nije potreban nacrt razmještaja procesora.**
  - **Jedan proces, nije potreba nacrt razmještaja procesa.**
  - **Vrlo mali programi, nije potreban nacrt implementacije.**
- **Neki dodatni pogledi:**
  - **Pogled podataka, pogled sigurnosti u sustavu**
- **Svaki pogled dokumentira se nacrtom – DIJAGRAMOM.**
- **Više pogleda (dijagrama) u okviru nekog konteksta = MODEL**
- **Model je apstraktan (reduciran, pojednostavljen, smanjen, ograničen) opis sustava iz određene perspektive.**

UML dijagrami

- Pojedini dijagram je pogled u model
  - Prikazan s polazišta određenog dionika
  - Daje određeno predstavljanje sustava
  - Semantički je dosljedan s ostalim pogledima
- U okviru UML v1.3 postoji devet standardnih dijagrama:
  - Statički pogledi: obrazaca uporabe, razreda, objekata, komponenti, razmještaja
  - Dinamički pogledi: slijeda, komunikacije, stanja, aktivnosti
  - **U RUP procesu svakoj aktivnosti pridružen je jedan ili više modela za opis, koji su dokumentirani s jednim ili više dijagrama (pogleda, *engl. views*).**

Percepcija UML-a



Potreba

- Osmisliti jezik koji osigurava jednostavan rječnik i pravila kombiniranja riječi u svrhu komunikacije.
- U jeziku modeliranja rječnik i pravila su usmjerena na konceptualnu i fizičku reprezentaciju sustava.
- UML standard
  - ”A language provides a *vocabulary* and *the rules for combining words* […] for the purpose of *communication*. A *modeling* language is a language whose vocabulary and rules focus on the conceptual and physical representation of a system. A modeling language such as the UML is thus a standard language for *software blueprints*.”

From ”UML user guide”

Osnovna svojstva

- Osnovne ideje:
  - obuhvatiti i opisati poslovne procese
  - poboljšati komunikaciju
  - pomoć u borbi s kompleksnošću
  - definirati logičku arhitekturu sustava omogućiti ponovno korištenje
- Namjena:
  - vizualizacija
  - specificiranje
  - konstrukcija
  - dokumentiranje

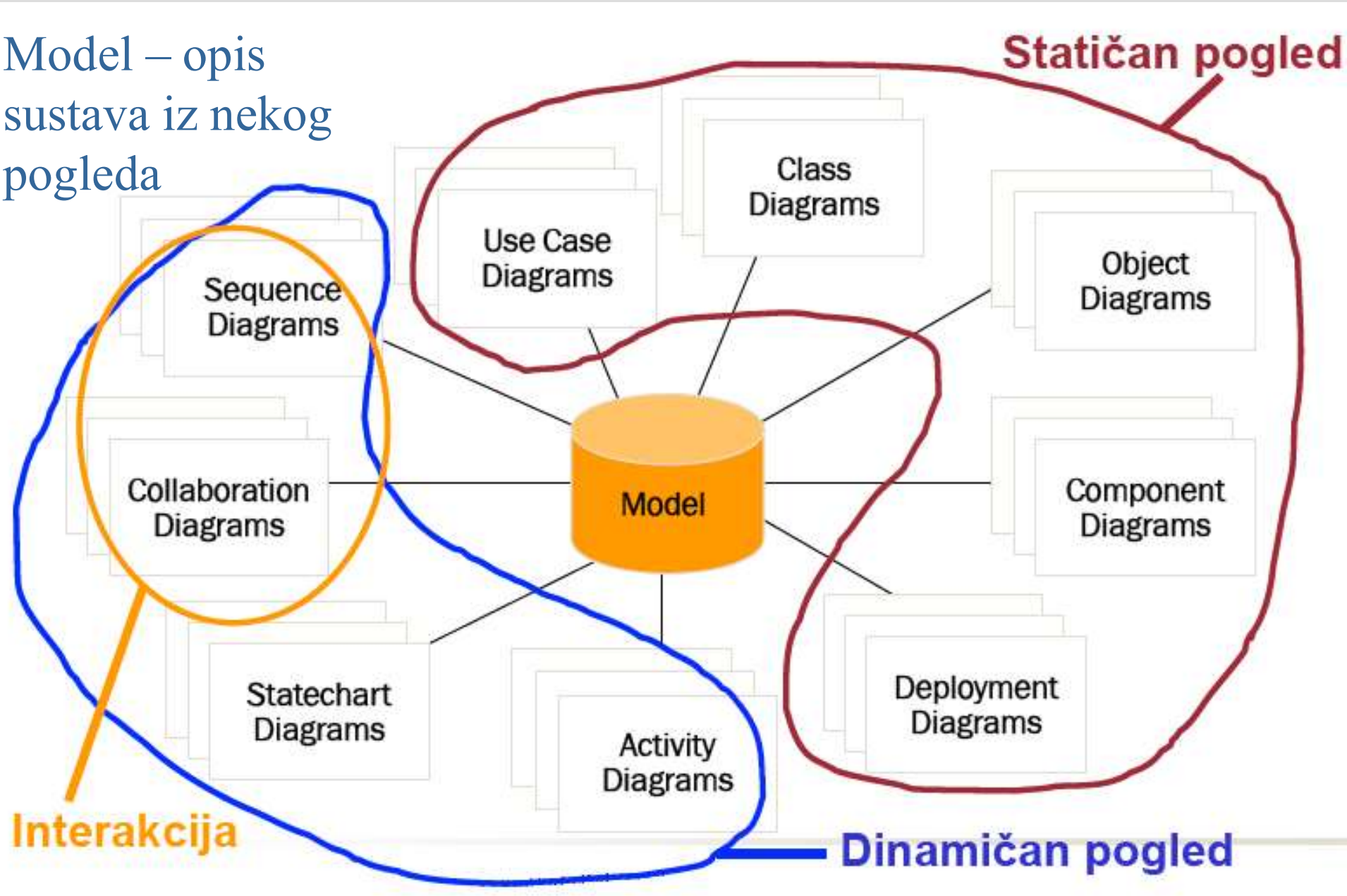
Osnovni elementi UML-a

- Stvari (*engl. things*)
  - strukturne stvari (*engl. structural things*)
    - *razred; sučelje; suradnja; obrasci uporabe; aktivni razred; komponenta; čvor*
    - aktori, signali, procesi i niti, aplikacije, dokumenti, datoteke, biblioteke, stranice, tablice
  - stvari ponašanja (*engl. behavioral things*)
    - interakcija; stanje
  - stvari grupiranja (*engl. grouping things*)
    - organizacija elemenata u grupe (samo konceptualno, hijerarhija), paketi
  - stvari označavanja (*engl. annotation things*)
    - opisni elementi, formalni/neformalni opis
- Relacije (*engl. relationships*)
  - Pridruživanja *(eng. association)*
  - Poopćenja *(eng. generalization)*
  - Realizacije *(eng. realization)*
  - Ovisnosti *(eng. dependency)*
- Dijagrami (*engl. diagrams*)
  - Dijagram razreda; objekata; slučajeva korištenja; toka; suradnji; stanja; aktivnosti; komponenata;....

Tri vrste pogleda (dijagrama)

Modeli, pogledi, dijagrami

Model – opis sustava iz nekog pogleda



Dijagrami

- Pogled na model
  - Iz perspektive dionika
  - Djelomična reprezentacija sustava
  - Semantički dosljedan s ostalim pogledima
- Standardni dijagrami (UML 1.1)
  - statični: dijagram obrazaca uporabe, dijagram razreda, dijagram objekata, dijagram komponenata, dijagram razmještaja
  - dinamični: dijagram slijeda, komunikacijski dijagram, dijagram stanja (“statechart”), dijagram aktivnosti
  - uz manje promjene najčešće upotrebljavani dijagrami i u novijim inačicama UML-a
- Specifičnost dijagrama
  - Svaki ima vlastitu sintaksu i semantiku ☹ ?
- Evolucija UML-a
  - veljača 2009. UML 2.2
  - ožujak 2011. UML 2.4-beta
  - → standard 2.5

Kada koristimo određene vrste dijagrama?

- **U izradi projekata i projektne dokumentacija** naglasak je najčešće na sljedećim dijagramima:
  - **dijagram obrazaca uporabe**
  - **dijagram slijeda**
  - **dijagram aktivnosti**
  - **dijagram razreda**
- **Za rješavanje arhitekture sustava** koriste se i drugi dijagrami:
  - **dijagram objekata, dijagram stanja, dijagram aktivnosti, komunikacijski dijagram, dijagram komponenti**
- **U implementacijskom dijelu dokumentacije** koristi se i **dijagram razmještaja**

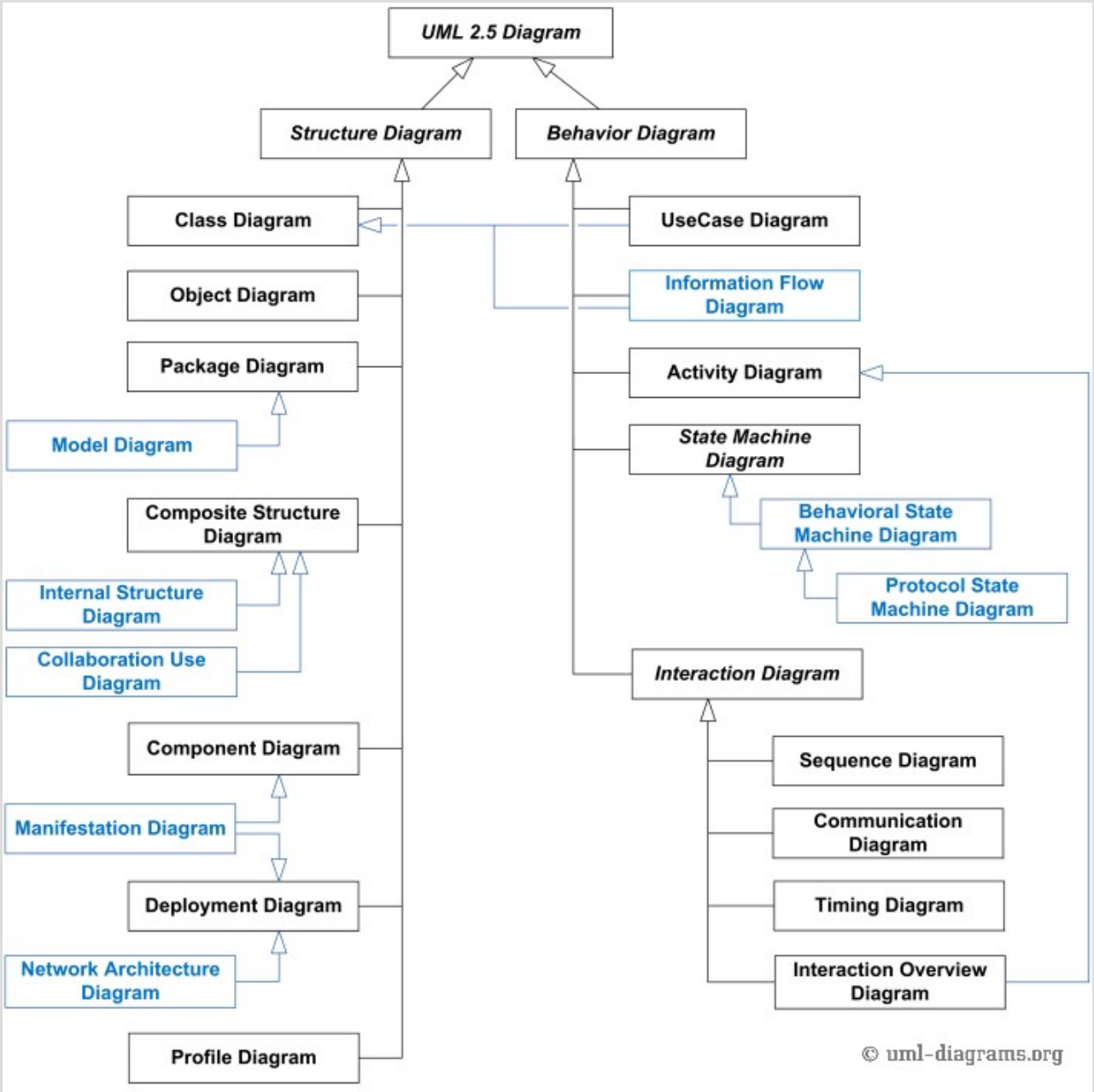
UML CASE-alati

- Enterprise Architect
- Visual Paradigm
- ArgoUML
- Astah Community
- plantUML
- Draw.io
- StarUML
- MS Visio
- …

UML Standard

- Trenutno: 2.5.1
- Specifikacija: https://www.omg.org/spec/UML/2.5.1/PDF (796 stranica)

Vrste dijagrama



Enterprise Architect

- https://sparxsystems.com/

Enterprise Architect

Firstly... What is UML?

The Object Management Group (OMG) specification states:

> *"The Unified Modeling Language (UML) is a graphical language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system. The UML offers a standard way to write a system's blueprints, including conceptual things such as business processes and system functions as well as concrete things such as programming language statements, database schemas, and reusable software components."*

The important point to note here is that UML is a 'language' for specifying and not a method or procedure. The UML is used to define a software system; to detail the artifacts in the system, to document and construct - it is the language that the blueprint is written in. The UML may be used in a variety of ways to support a software development methodology (such as the Rational Unified Process) - but in itself it does not specify that methodology or process.

UML defines the notation and semantics for the following domains:

- The User Interaction or Use Case Model - describes the boundary and interaction between the system and users. Corresponds in some respects to a requirements model.
- The Interaction or Communication Model - describes how objects in the system will interact with each other to get work done.
- The State or Dynamic Model - State charts describe the states or conditions that classes assume over time. Activity graphs describe the workflows the system will implement.
- The Logical or Class Model - describes the classes and objects that will make up the system.
- The Physical Component Model - describes the software (and sometimes hardware components) that make up the system.
- The Physical Deployment Model - describes the physical architecture and the deployment of components on that hardware architecture.

https://sparxsystems.com/resources/tutorials/uml/part1.html

The UML also defines extension mechanisms for extending the UML to meet specialized needs (for example Business Process Modeling extensions).

Part 2 of this tutorial expands on how you use the UML to define and build actual systems.

REFERENCE I LITERATURA

- A. Jović, N. Frid, D. Ivošević: Procesi programskog inženjerstva, 3. izdanje, Sveučilište u Zagrebu, FER ZEMRIS, 2019.
- I. Sommerville, Software engineering, 8th ed., Addison Wesley, 2007.
- G. Overgaard, B. Selic, C. Bock, OMG, 2000.
- S. Tockey: How to Engineer Software, Wiley-IEEE Computer Society Press, 2019.

Hvala na pažnji!

Pitanja?