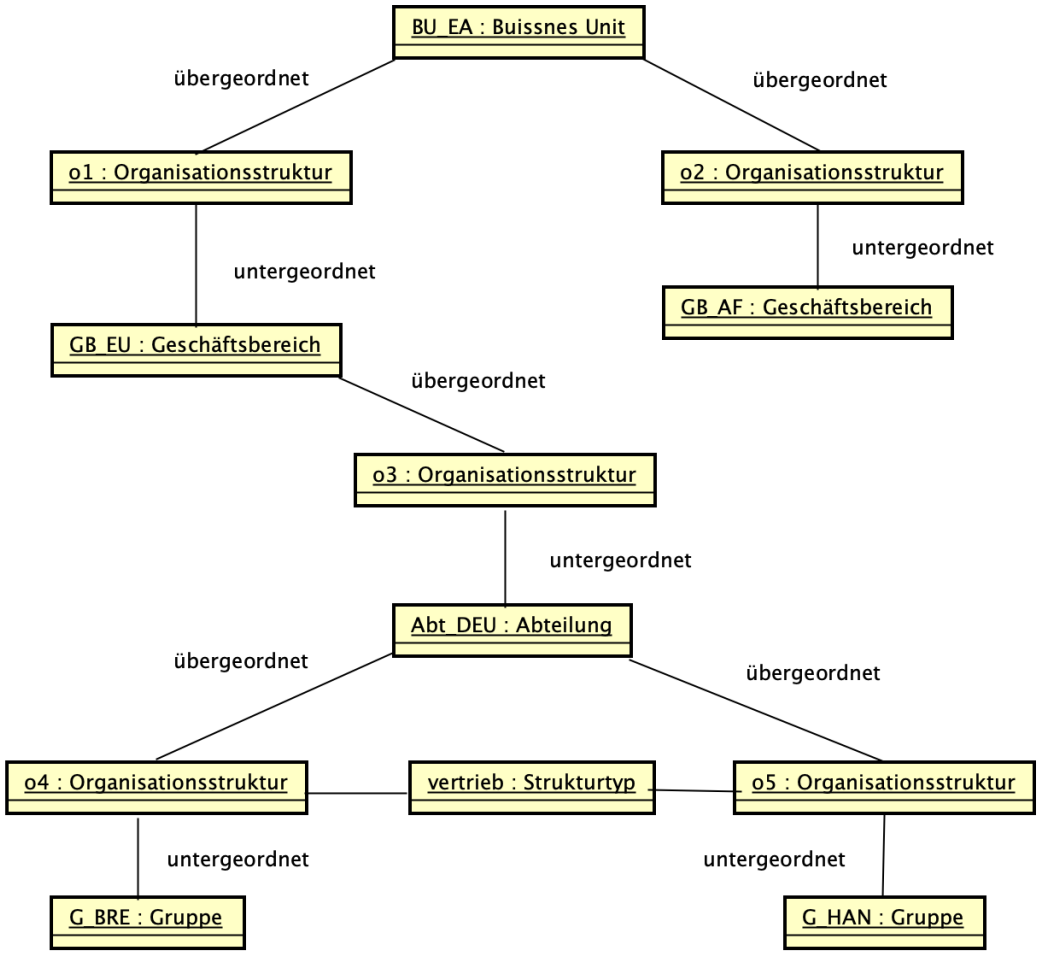
Furkan Hidayet Rafet Aydin
1630039

Aufgabe 1: Organisationshierarchie-Pattern
In Kapitel 2.3 wurde das Organisationshierarchie-Pattern anhand der Modellierung der Organisationsstruktur eines Unternehmens eingeführt.
Zeichnen Sie ein UML-**Objektdiagramm** (also eine Instanz des Klassendiagramms aus **Folie 20** mit Objekt–Instanzen und deren Beziehungen zueinander), das die folgenden Beziehungen darstellt:

- Business Unit Europa_Afrika (BU_EA) *ist weisungsbefugt für* Geschäftsbereich Europa (GB_EU).
- Business Unit Europa_Afrika (BU_EA) *ist weisungsbefugt für* Geschäftsbereich Afrika (GB_AF).
- Geschäftsbereich Europa (GB_EU) *ist weisungsbefugt für* Abteilung Deutschland (Abt_DEU).
- Gruppe Hannover (G_HAN) *gehört vertrieblich zu* Abteilung (Abt_DEU).
- Gruppe Bremen (G_BRE) *gehört vertrieblich zu* Abteilung (Abt_DEU).

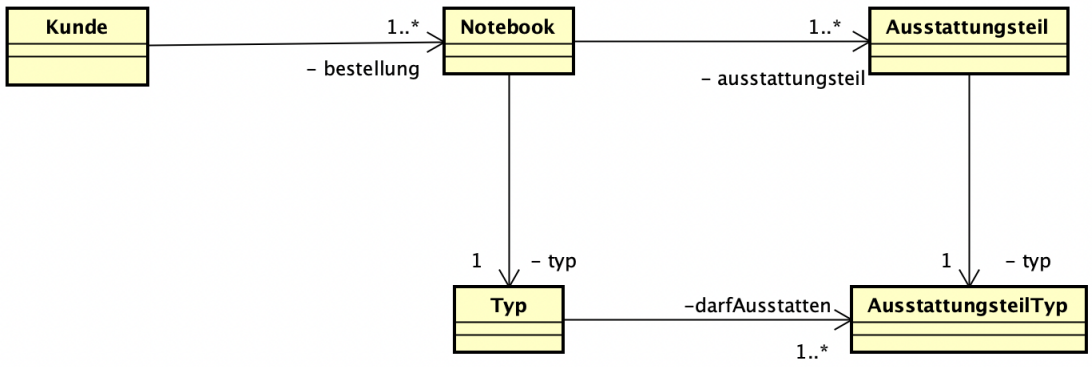


Aufgabe 2: Wissens-Ebene-Pattern (Pflichtaufgabe)
Entwerfen Sie ein Klassendiagramm für ein Software-System zur Bestellung von Notebooks, das den folgenden Sachverhalt abbildet:

- Ein Kunde kann ein oder mehrere Notebooks bestellen.
- Jedes Notebook ist von einem bestimmten Typ (z.B. ThinkPad T490, Apple MacBook Pro, Acer Swift etc.)
- Für jedes Notebook kann eine Menge von Ausstattungsteilen ausgewählt werden (z.B. Grafikkarte, Hauptspeichererweiterung, Stromkabel etc.)
- Jedes Ausstattungsteil ist von einem bestimmten Typ.
- Neue Typen für Notebooks als auch für Ausstattungsteile sollen dynamisch zur Laufzeit definiert werden können.

(a) Entwerfen Sie ein Klassenmodell mit allen Beziehungen und Kardinalitäten und benennen Sie jede Beziehung mit einem geeigneten Namen.

(b) Erweitern Sie das Klassenmodell um eine Metaebene, so dass Ausstattungsteile eines bestimmten Typs nur für bestimmte Notebooks des richtigen Notebook-Typs gewählt werden können (z.B. Lenovo-ThinkPad Pro Docking Station für ThinkPad T490, aber nicht für Apple MacBooks).



Aufgabe 3: Implementierung des Wissens-Ebene-Pattern (Pflichtaufgabe)
Erweitern Sie Ihr Klassenmodell aus Aufgabe 2 um die notwendigen Methoden zum Pflegen der Beziehungen. Programmieren Sie die Klassen aus und schreiben Sie ein Hauptprogramm, das Ihre Klassen testet und unterschiedlichen Notebooks unterschiedliche Ausstattungsteile zuordnet.

```java
public class Kunde {

    ArrayList<Notebook> bestellung = new ArrayList<>();

    public Kunde() {}

    public void addNotebook(Notebook notebook) {
        bestellung.add(notebook);
    }

    public void removeNotebook( Notebook notebook) {
        bestellung.remove(notebook);
    }

    public ArrayList getBestellung() {
        return bestellung;
    }

    public ArrayList setBestellung(ArrayList<Notebook> bestellung) {
        this.bestellung = bestellung;
    }
}
```

```java
public Notebook(String name, Notebooktype typ) {
    this.name = name;
    this.typ = typ;
}

public void addAusstattung(Ausstattungsteil ausstattungsTeil) {
    if(this.getType().isAllowedToAdd(ausstattungsTeil.getType())) {
        this.ausstattung.add(ausstattungTeil);
        System.out.println(x: "Ausstattung Hinzugefuegt");
    } else {
        System.out.println(x: "Konnte nicht hinzugefuegt werden");
    }
}

public void removeAusstattung(Ausstattungsteil ausstattungsTeil) {
    this.ausstattung.remove(ausstattungTeil);
}

public String getName() {
    return this.name;
}

public void setName(String name) {
    this.name = name;
}

public ArrayList<Ausstattungsteil> getAusstatung() {
    return this.ausstatung;
}

public void setAusstattungsteilTyp(ArrayList<Ausstattung> ausstattung) {
    this.ausstattung = ausstattung;
}

public Notebooktype getType() {
    return this.typ;
}
```

```java
public class Notebooktype {
    private String name;
    ArrayList<AusstattungsteilTyp> ausstattungsTypen = new ArrayList<>();

    public Notebooktype(String name) {
        this.name = name;
    }

    public boolean isAllowedToAdd(AusstattungsteilTyp ausstattungsteilTyp) {
        return ausstattungsTypen.contains(ausstattungsteilTyp);
    }

    public void addAusstattungstyp(AusstattungsteilTyp ausstattungsteilTyp) {
        this.ausstattungsTypen.add(ausstattungsteilTyp);
    }

    public void removeAusstattungstyp(AusstattungsteilTyp ausstattungsteilTyp) {
        this.ausstattungsTypen.remove(ausstattungsteilTyp);
    }

    public String getName() {
        return this.name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public ArrayList<AusstattungsteilTyp> getAusstatungstypen() {
        return this.ausstattungsTypen;
    }

    public void setAusstattungstypen(ArrayList<AusstattungsteilTyp> ausstattungsTypen) {
        this.ausstattungsTypen = ausstattungsTypen;
    }
}
```

```java
public class Ausstattungsteil {
    String name;
    AusstattungsteilTyp ausstattungsteilTyp;

    public Ausstattungsteil(String name, AusstattungsteilTyp ausstatungsteilTyp) {
        this.name = name;
        this.ausstattungsteilTyp = ausstatungsteilTyp;
    }

    public String getName() {
        return this.name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public AusstattungsteilTyp getTyp() {
        return ausstattungsteilTyp;
    }

    public void setTyp(AusstattungsteilTyp ausstattungsteilTyp) {
        this.ausstattungsteilTyp = ausstattungsteilTyp;
    }
}
```

```java
public class AusstattungsteilTyp {
    String name;

    public AusstattungsteilTyp(String name) {
        this.name = name;
    }

    public String getName() {
        return this.name;
    }

    public void setName(String name) {
        this.name = name;
    }
}
```

Run | Debug

```java
public static void main(String[] args) {
    Notebooktype MacBookPro = new Notebooktype(name: "MacBookPro");

    AusstattungsteilTyp gpu = new AusstattungsteilTyp(name: "gpu");
    AusstattungsteilTyp cpu = new AusstattungsteilTyp(name: "cpu");

    MacBookPro.addAusstattungstyp(gpu);

    Notebook mac = new Notebook(name: "mac", MacBookPro);

    Ausstattungsteil a = new Ausstattungsteil(name: "a", gpu);
    Ausstattungsteil b = new Ausstattungsteil(name: "b", cpu);

    mac.addAusstattung(a);
    mac.addAusstattung(b);
}
```