קורס הנדסת תוכנה – סמסטר א' 2020 מדעי המחשב
נכתב ע"י צבי מינץ, מבוסס על מצגות והתרגולים של סגל הקורס
גב' **ספיר אסרף**, גב' **שקרון מירב** וגב' **אביגל שטקל**

דיאגרמות – הנדסת תוכנה

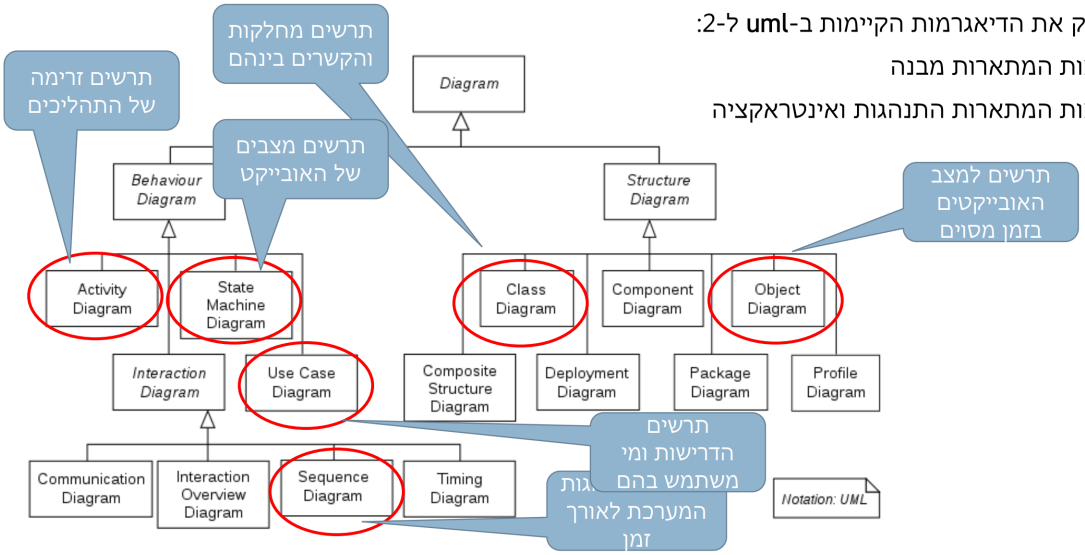
UML

שפת UML - שפת התרשימים UML מאפשרת למדל **מערכות תוכנה** באמצעות **תרשימים גרפיים**.
החשיבות שיש לאפשרות ליצור מידול גרפי לפני שמושקעים המשאבים ומתחיל תהליך הפיתוח דומה לחשיבות שיש ליצירת מודל פיזי קטן על ידי ארכיטקט לפני שמתחילים בבניה בפועל.
השפה כוללת סה"כ 13 סוגים שונים של דיאגרמות, כאשר כל אחת מהן מתארת היבט אחר של המערכת המתכוננת

מבט על:

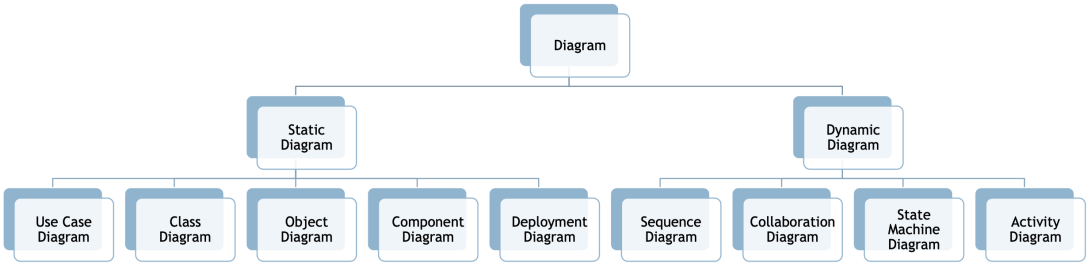
ניתן לחלק את הדיאגרמות הקיימות ב-uml ל-2:

- דיאגרמות המתארות מבנה
- דיאגרמות המתארות התנהגות ואינטראקציה



חלוקה נוספת:

- דיאגרמות סטטיות
- דיאגרמות דינמיות



עקרונות:

אחידות – שפה גרפית עם שיטת סימון יחידה.
פשטות – למרות שהיא מכילה הרבה סימונים ודיאגרמות, האחידות יוצרת פשטות – קל להבין את המערכת לאחר הסתכלות בתרשים הגרפי.
פרויקטים גדולים – מיועד בעיקר לפרויקטים גדולים
אינה תלויה בטכנולוגיה – אין תלות בחומרים או בתוכנה
מתאימה לכולם – בין אם לכתובי קוד, ללקוח, לתכוניתן, למעצב או למנהל הפרויקטים – כל אחד יכול להבין את התרשימים הגרפיים

Tradeoff

יתרונות	חסרונות
1. התמקדות בתחום הידע של המשתמש והתאמה לדרך החשיבה שלו 2. תרשימים גרפיים ידידותיים למשתמש, למפתח, למנהל הפרויקט וכו' 3. חוסר תלות בטכנולוגיה	1. ריבוי מודלים מכביד על תהליך הפיתוח 2. הקשר בין המודלים מורכב 3. קושי בשימוש במערכות מורכבות כי קשה למדל אותם

סוגי דיאגרמות:

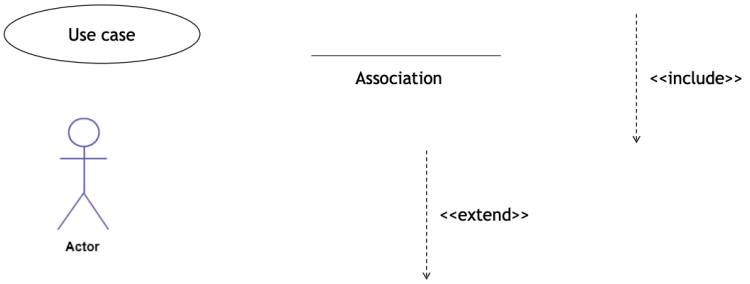
Use Case Diagram .1

הסבר: במילים אחרות, דיאגרמת אופן השימוש
סוג של דיאגרמה **התנהגותית** אשר מורכבת מניתוח תרחישי שימוש.

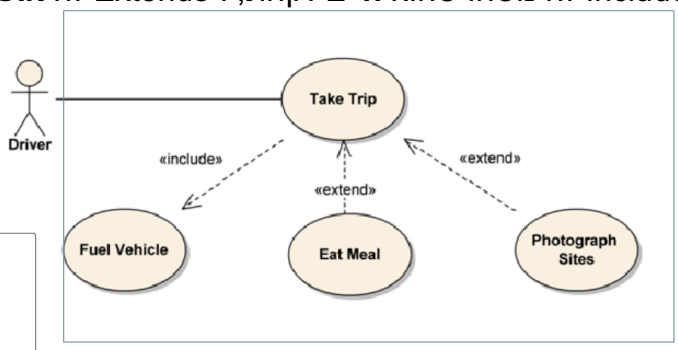
- מתארת את הפעולות שהמערכת מבצעת ויש להם תוצאות גלויות.
- מראה את האינטראקציה בין דברים מחוץ למערכת למערכת עצמה.
- מודל יכול להתייחס למערכת כולה או לחלקה.
- המודל לא מראה את סדר הדברים- לא סדר כרונולוגי ולא על ציר הזמן.

מטרה: מטרת הדיאגרמה היא הצגת רקע פונקציונאלי של המערכת עם שימוש במושגים:
שחקנים, והתלות בין תרחישי השימוש.
המטרה **המרכזית** היא הצגת הפעולות המבוצעות על ידי כל שחקן. תפקידי השחקן ישורטטו בתרשים.

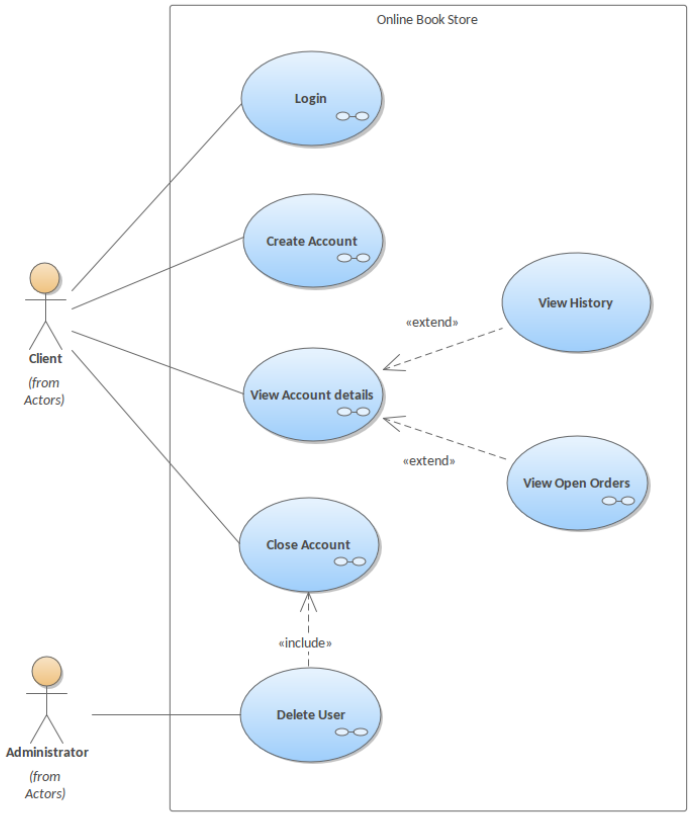
שיטות סימון



כאשר:
Include זה משהו שהוא **חייב** לקחת, ו-Extends זה **אופציונאלי** (לא חייב)



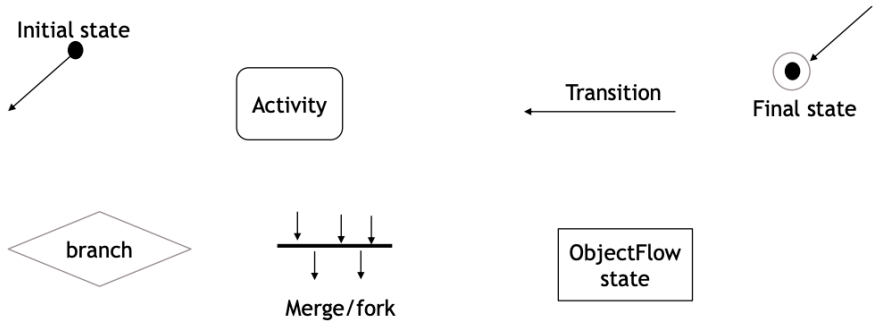
דוגמא:



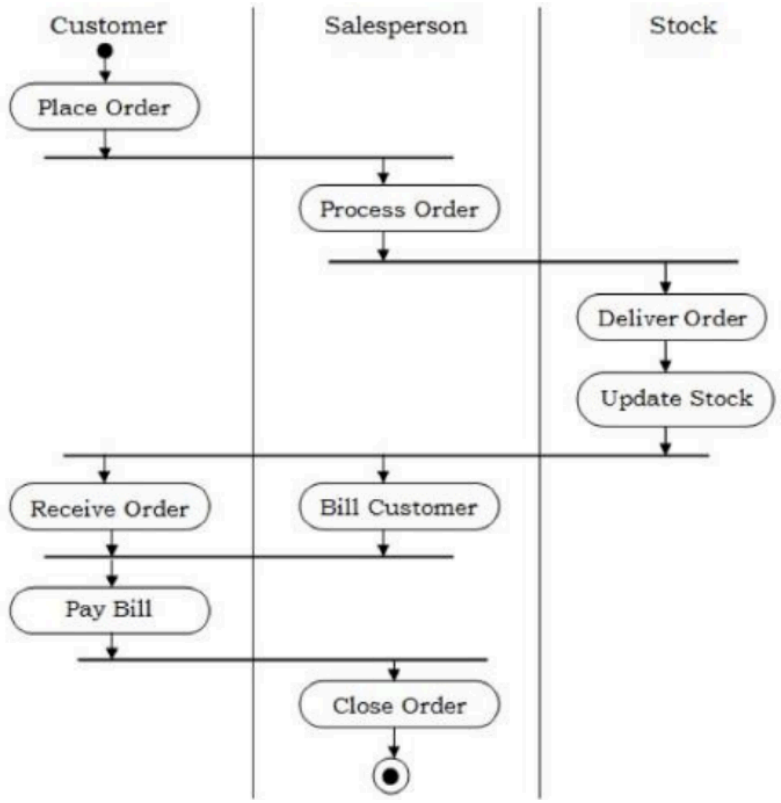
Activity Diagram .2

הסבר: תרשים פעילות מציג את הזרימה **מפעילות לפעילות** בתחום המערכת ובין השחקנים
כל פעילות ב-Use case נהפכת להיות תרשים פעילות אחד

שיטות סימון:



דוגמא:

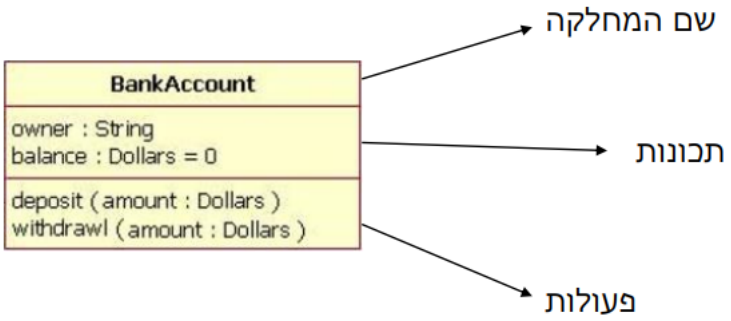


Class Diagram .3

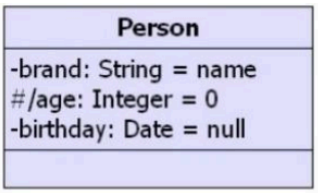
הסבר: תרשים סטטי המתאר את מבנה המערכת על ידי הצגת מחלקותיה, תכונותיהן והקשרים בין המחלקות.

- מודל של מבנה המערכת ע"י מידול המחלקות, התכונות והפעולות.
- UML class diagram הוא תכנית האב של המחלקות הדרושות לבניית התוכנה
- מתכנתים מפתחים את המערכת בהתבסס על מודל המחלקות שהוגדר
- זהו המודל הנפוץ ביותר ב-UML

מטרה: מטרת הדיאגרמה
שיטות סימון:
סימון מחלקה:



ערכים במחלקה:

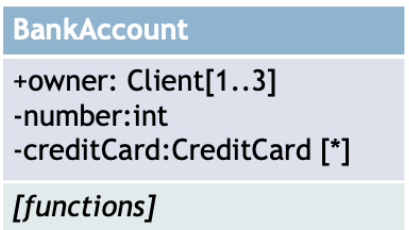


- Private -
- Public +
- Protected #
- Package ~

- הסימן / מתאר מצב שבו ערכו של המשתנה המתואר הוא Attribute Derived
- ניתן לסמן כל משתנה מה יהיה הערך ה-default שלו אחרי השיוויון

בנוסף ניתן לציין את כמות המופעים למשתני מחלקה באופן הבא:

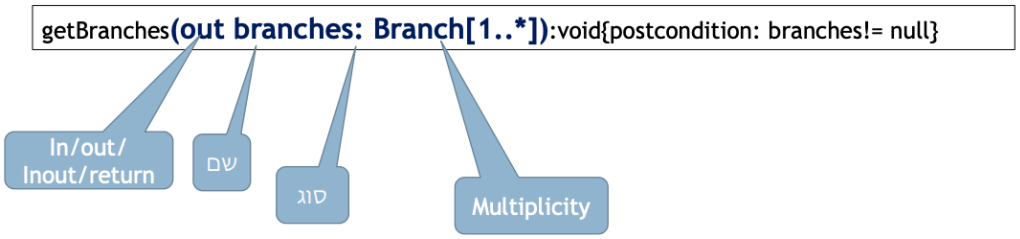
- מייד לאחר ציון ה- type של המשתנה ניתן לציין את ה- multiplicity שלו.
- את ה- multiplicity מציינים באמצעות מתן טווח ערכים אפשרי.
- כאשר לא מציינים את ה-multiplicity של משתנה מסויים ברירת המחדל היא 1.
- כאשר משתמשים בסימן * כדי לציין את ה-multiplicity אז המשמעות היא שהמשתנה האמור ישמש לייצוג ערך אחד או יותר (עד אינסוף)



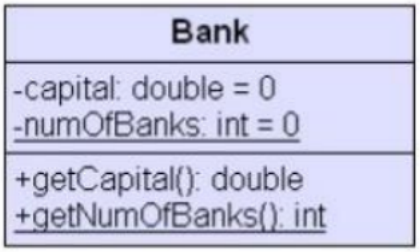
מתודות בתרשים:

- הפורמט להגדרת מתודה

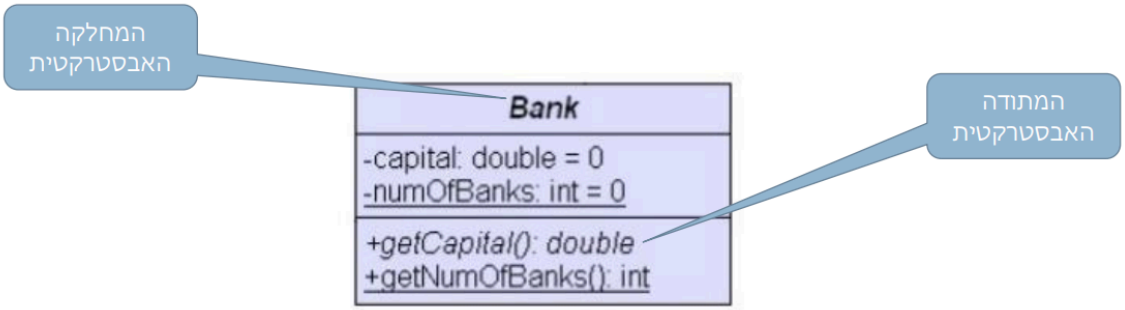
- הגדרת הפרמטרים-



הערה: מתודות סטטיות יסומנו עם קו תחתון, באופן הבא:

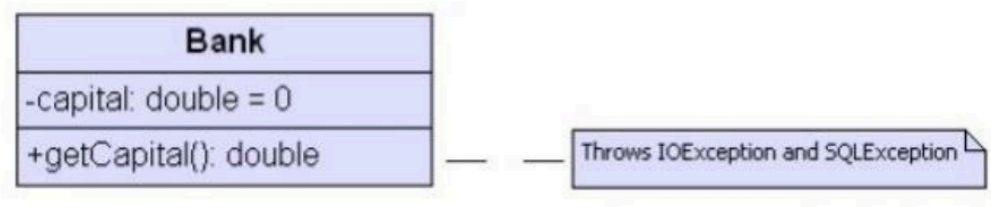


הערה: מחלקות אבסטקטיות יהיו נטיות, באופן הבא:

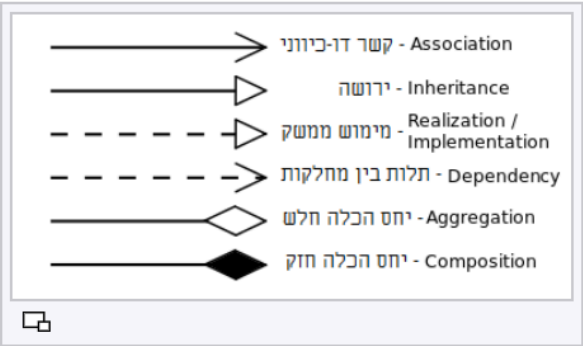


זריק

- כאשר בקריאה להפעלת מתודה יש סכנה שתתרחש תקלה (ייזרק exception) מקובל להוסיף note ובו תיאור ה- exception שעלול להיזרק ולחברו בקו מקווקו למתודה שבה מדובר.



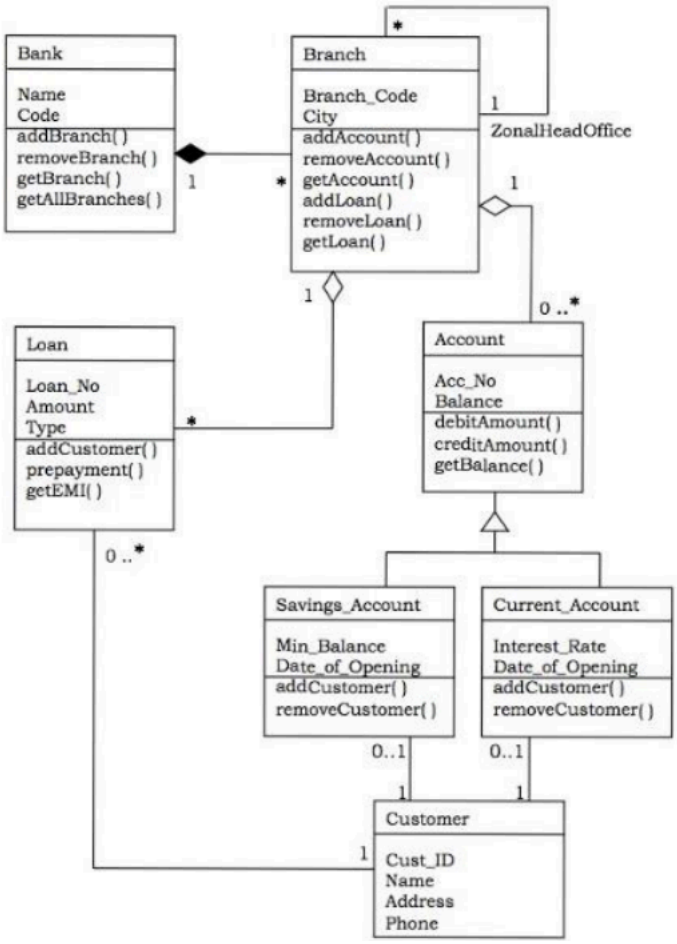
קשרים בין מחלקות:



היחסים הקיימים בדיאגרמת מחלקה הם:

- Association - יחס כלשהו בין מחלקות. המספרים בקצוות מציינים את מספר (או טווח) העצמים בכל צד של היחס (Multiplicity).
- Inheritance - ירושה ממחלקת בסיס.
- Realization / Implementation - מימוש ממשק.
- Dependency - תלות של מחלקה אחת במחלקה אחרת.
- Aggregation - יחס הכלה חלש: מחלקה מכילה מצביע לעצם, כאשר העצם המוצבע יכול להתקיים ללא המחלקה המכילה.
- Composite Aggregation - יחס הכלה חזק: מחלקה מכילה עצם, כאשר העצם המוכל מתקיים רק עם העצם החיצוני.

<u>דוגמא כוללת:</u>

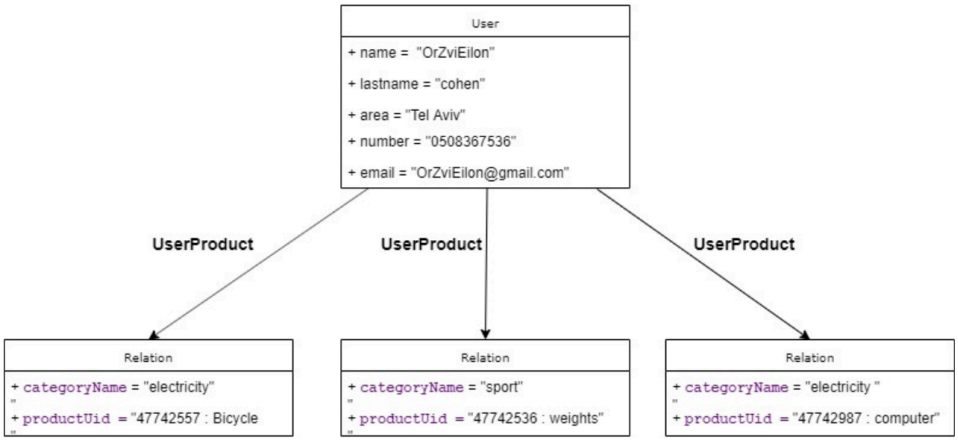


Object Diagram .4

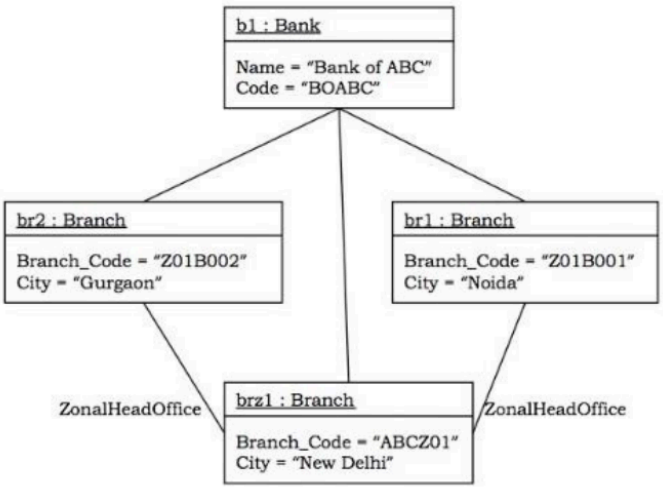
הסבר:

- דיאגרמת אובייקטים מתארת מופע של מודל המחלקות, בדומה לתרחישים אמיתיים שעל בסיסם בונים את המערכת.
- דיאגרמת אובייקטים היא מודל סטטי (בדומה למודל המחלקות)
- השימוש במודל האובייקטים דומה למודל המחלקות רק שהם מאפשרים בניית אב טיפוס של המערכת מנקודת מבט מעשית

דוגמא:



או לחלופין:



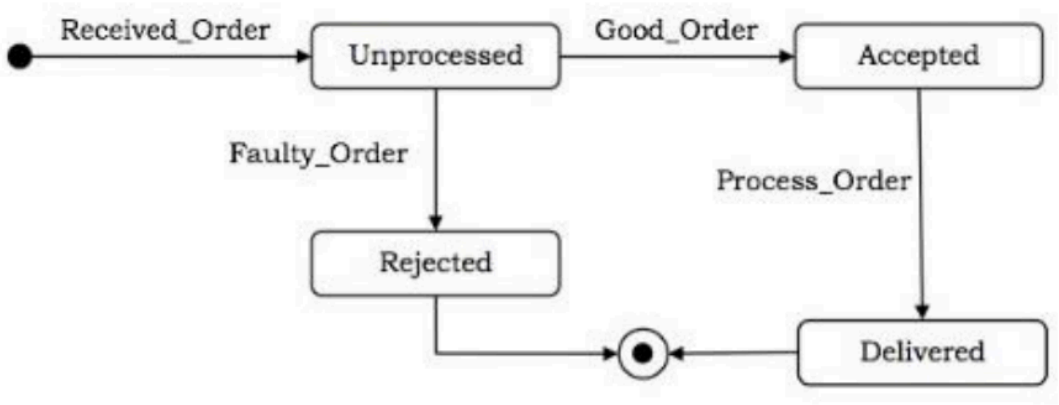
Trade Off

	Class diagram	Object diagram
מבנה	מכיל 3 חלקים: שם, רשימת תכונות ורשימת פעולות	מכיל 2 חלקים: שם ורשימת מאפיינים
שם המחלקה	שם המחלקה עומד בפני עצמו בחלק של שם המחלקה	הפורמט מורכב משם האובייקט + נקודותיים + שם המחלקה (Tom:Employee)
רשימת התכונות	מגדיר את התכונות של המחלקה	מגדיר את הערך הנוכחי של כל תכונה
רשימת הפעולות	כלולות	לא כלולות
קשרים	מוגדר הקשר בין מחלקות- שם הקשר וכמות הקשר.	מוגדר שם הקשר, אבל לא הכמות (לא רלוונטי כשמדברים על ישות בודדת)

State Machine Diagram .5

הסבר: בתרשים מצבים שמים את האובייקט במרכז, התרשים מתאר את זרימת האובייקטים ממצב אחד לאחר ואת המצבים השונים שבו האובייקט יכול להיות לאורך חיי המערכת
הערה: כל תרשים מתאר מחלקה אחת
שיטות סימון:

דוגמא:



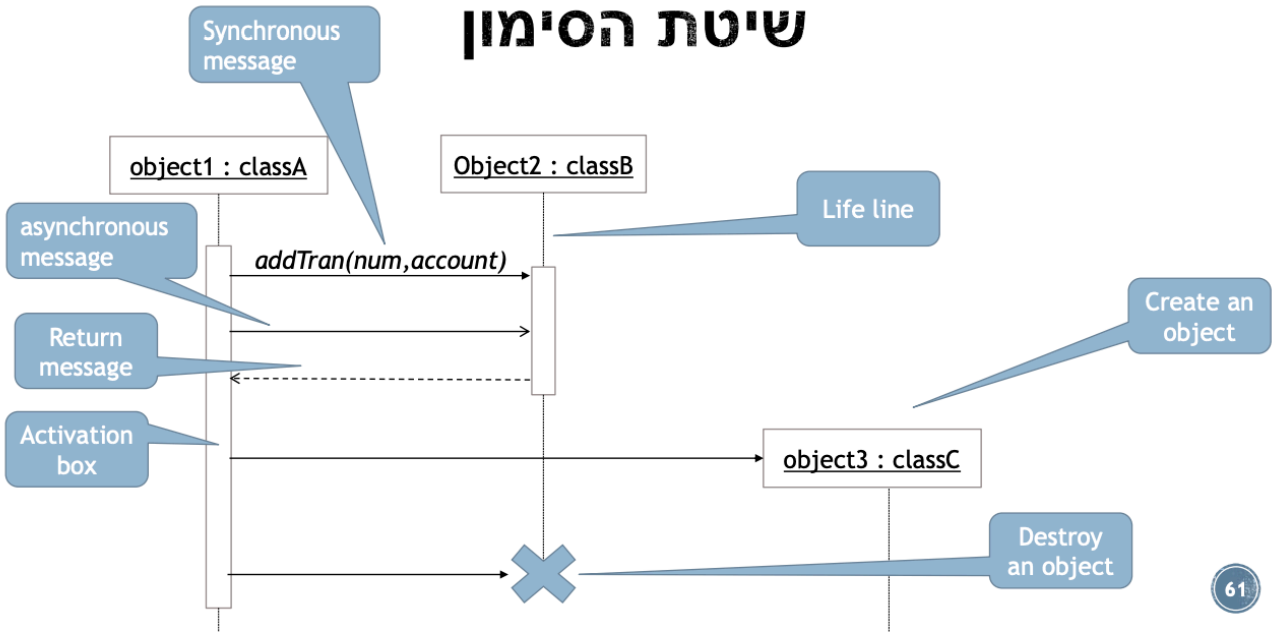
Sequence Diagram .6

הסבר:

- תרשים רצף הממחיש את סדר הפעילויות ברצף של הזמן.
- תרשים רשת נכתב בצורה של תרשים דו מימדי
  - על ציר ה-X נמצאים האובייקטים
  - על ציר ה-Y ממוקמות ההודעות שהאובייקטים האלה שולחים
- לכל פונקציונליות נכין תרשים רצף נפרד

מטרה: תיאור האופן שבו המערכת מבצעת תהליך.

שיטות סימון:

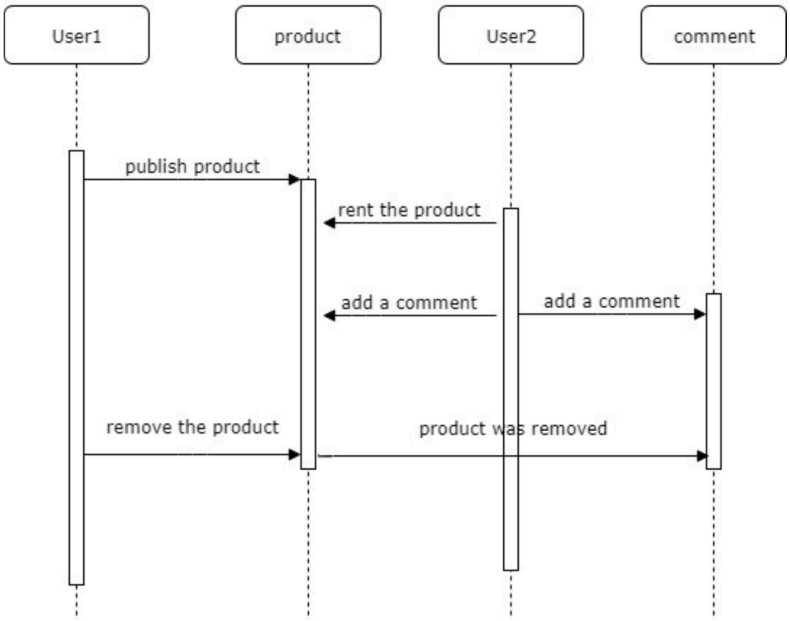


Tradeoff

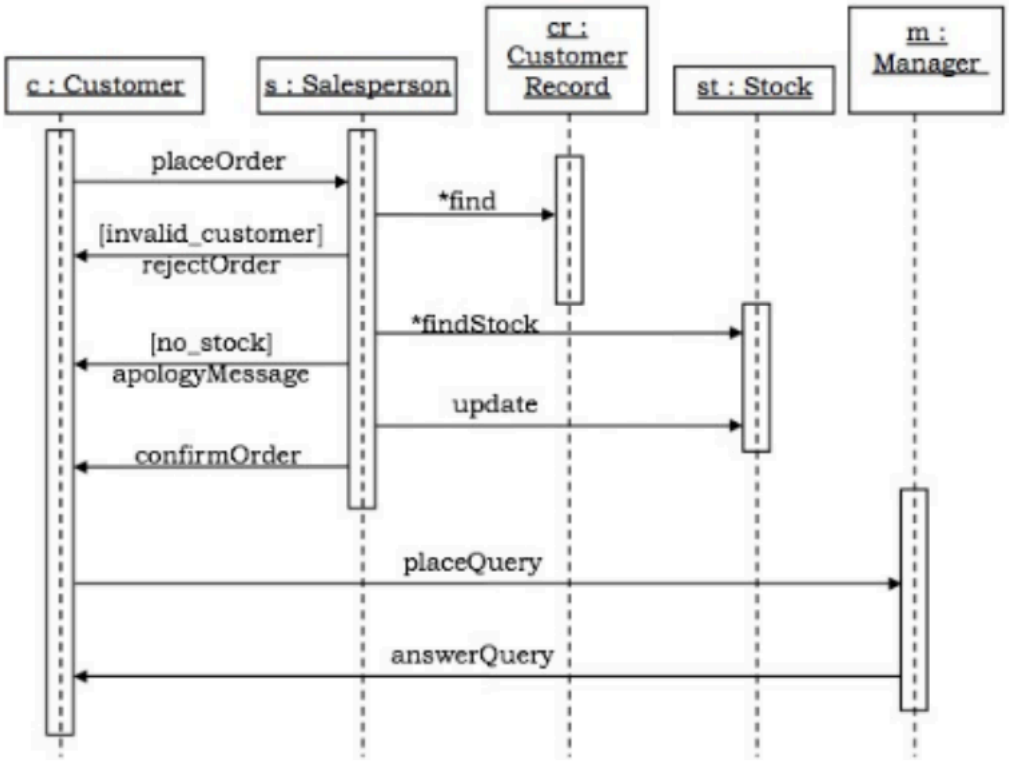
Sequence	Activity
דיאגרמת אינטראקציה	דיאגרמת התנהגות
מתמקד בהודעות שמועברות בין ישויות במערכת.	מתמקד בתהליך של האובייקטים ונותן דגש לרצף ולתנאים שיש בתהליך
נותן דגש לסדר ביצוע הפעולות	סדר ביצוע הפעולות לא מודגש
התהליך יותר מדויק ומפורט	כללי יותר ופחות מדויק

דיאגרמות – הנדסת תוכנה

דוגמא:



או לחלופין:



מה הסדר הנכון?

- אין!
- אבל הסדר ההגיוני הוא:
  - Use case diagram
  - State machine diagram
  - Activity diagram
  - Class diagram
  - Object diagram
  - Sequence diagram

ERD .7

האחראי:

בשנת 1976

הסבר: קיצור של **E**ntity **R**elationship **D**ata Model
זוהי בעצם דרך להציג מידע. הצגה כזאת מאפשרת תכנון מלמעלה-למטה של מערכות מסדי נתונים יחסיות. המודל אינו מתחשב בארכיטקטורת המחשב עליו יורץ מסד הנתונים אלא רק במבנה הלוגי הרצוי של מסד הנתונים באופן שיאפשר נוחות ויעילות בגישה למידע.

מבנה:

המודל מורכב מ-3 רכיבים בסיסיים:

- ישויות - Entities
- תכונות - Attributes
- קשרים - Relationships

להסברים מורחבים על המודל – להיכנס למודל

דוגמא: