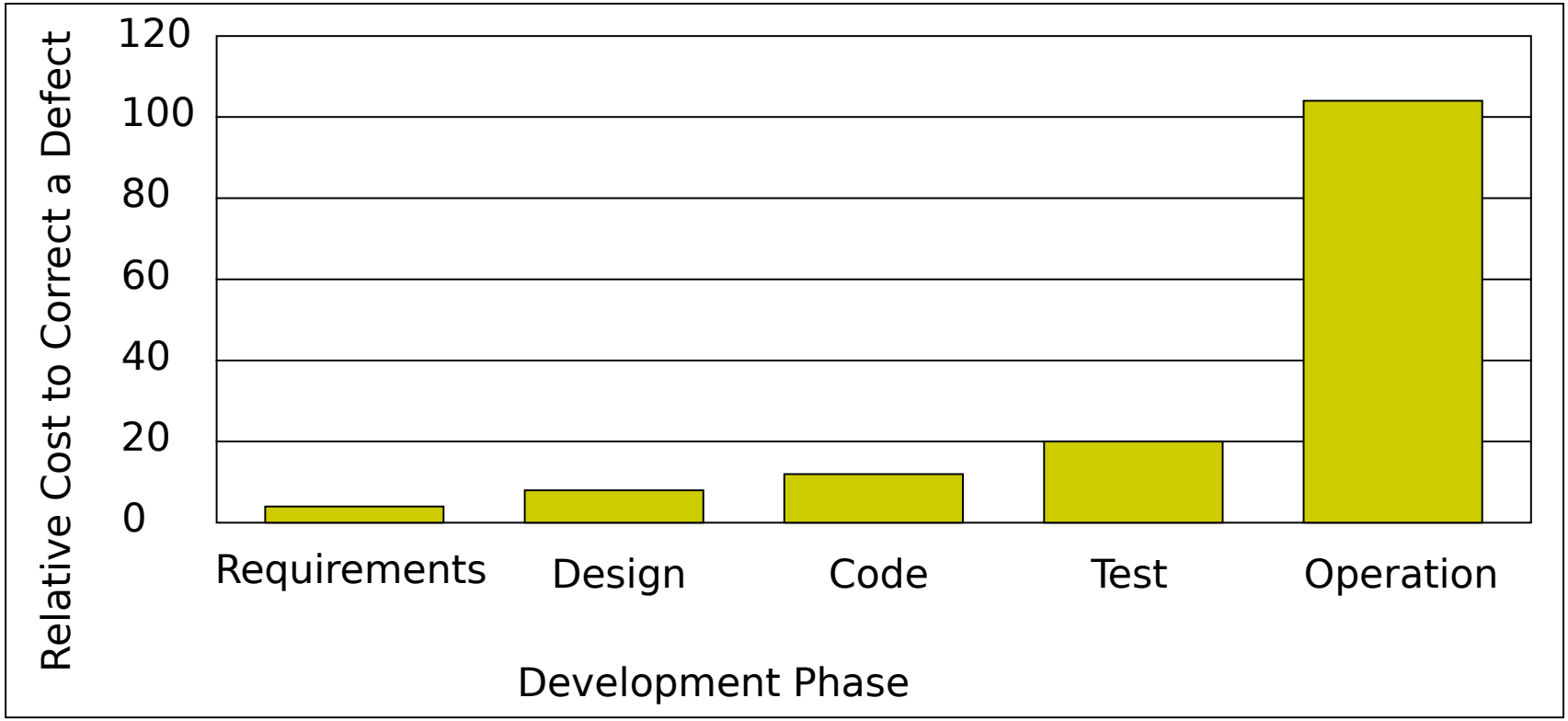
מבוא לתיכנות מונחה עצמים – תהליכי ניתוח ותיכנון

מקורות:

- Craig Larman, “Applying UML and Patterns”
- Dr. Rahmi Marasli, CMU course
- Dr. Mira Balaban, BGU course
- GWT Lecturer, “UML Tutorials”,
  https://www.youtube.com/watch?v=y7grsHY9Fa0&list=PLoWne5q-c9E_Q2_eAUZKPDA5K0V-O5zXs

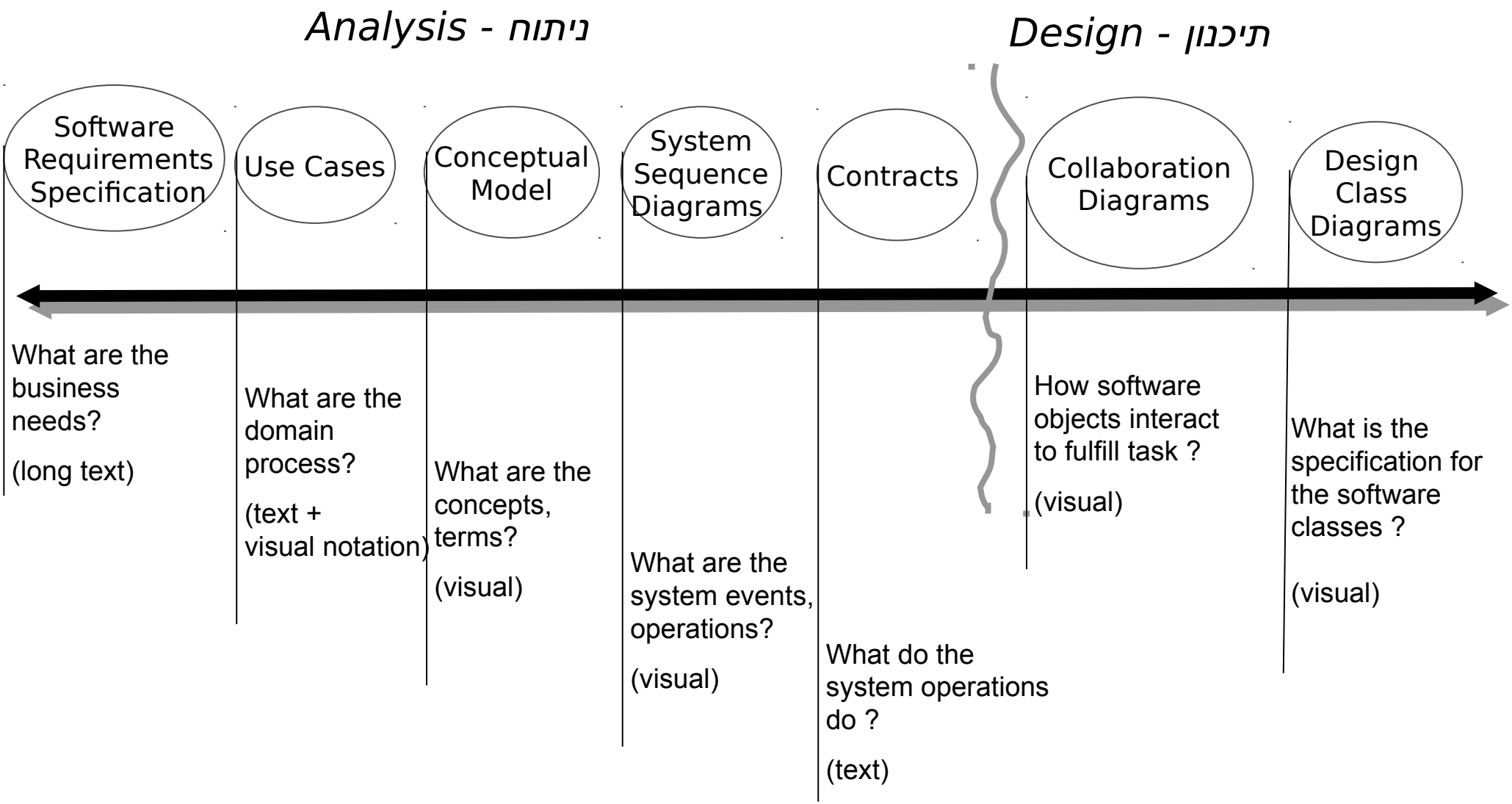
ניתוח ותיכנון – למה צריך את זה?



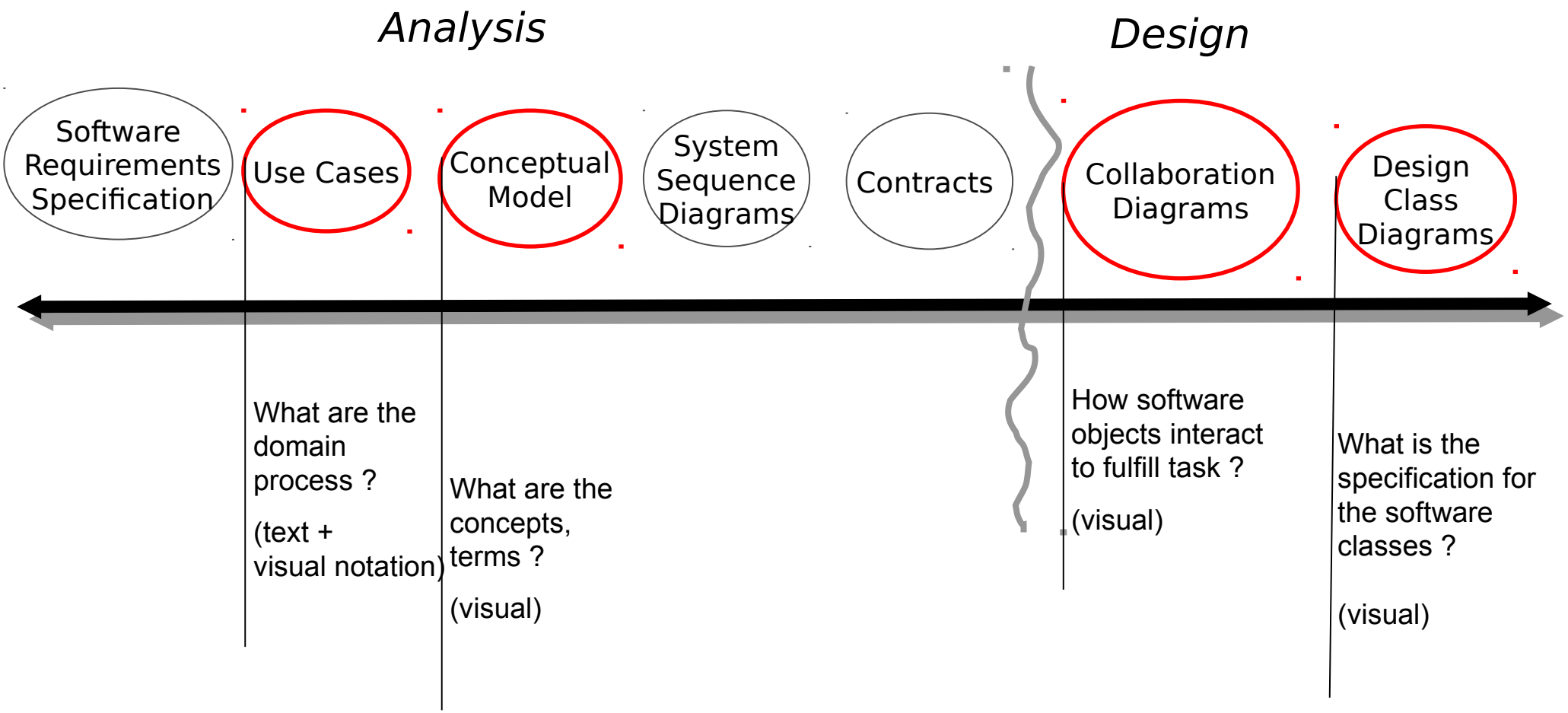
Grady 1999

מחזור-חיים של תוכנה

תהליכי ניתוח ותיכנון

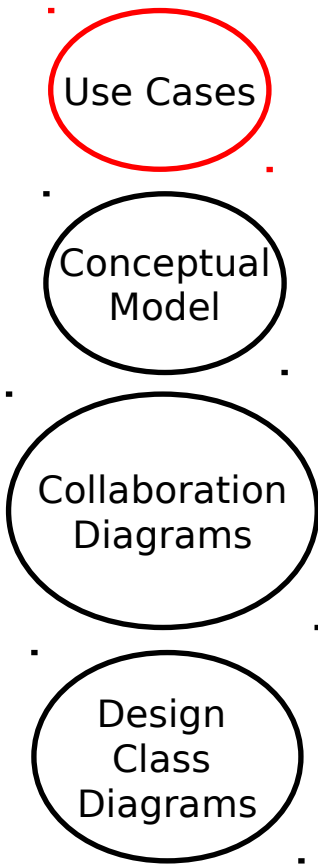


דוגמה – משחק קוביות



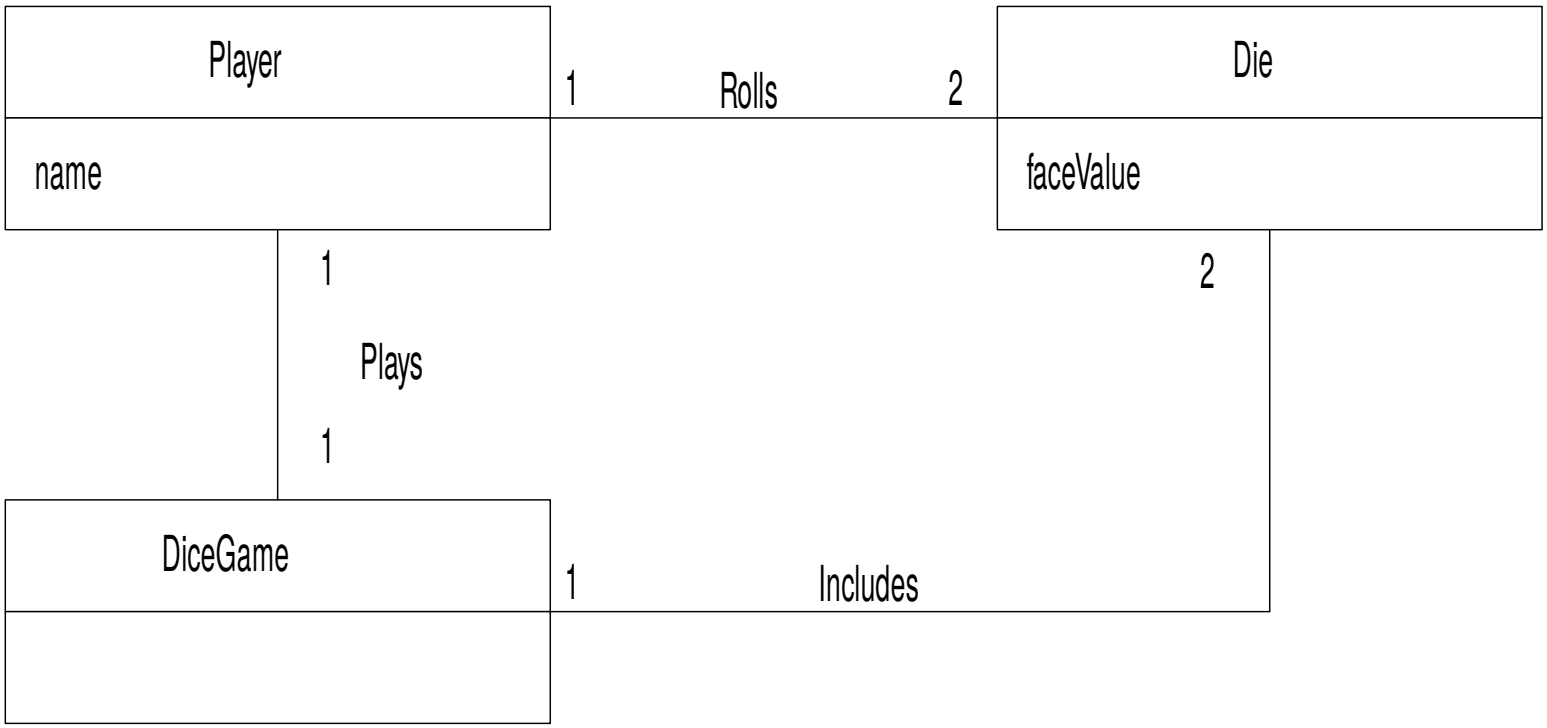
דוגמה – משחק קוביות
Use case – תרחיש שימוש

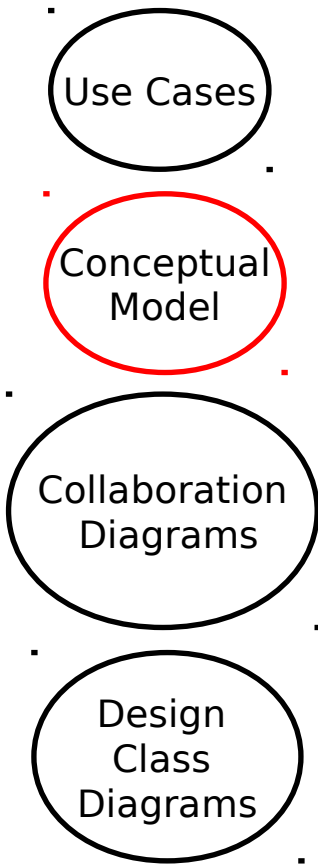
Use case:	Play a Game
Actors:	Player
Description:	This use case begins when the player picks up and rolls the dice. If the dice total seven he wins; otherwise – he loses



דוגמה – משחק קוביות

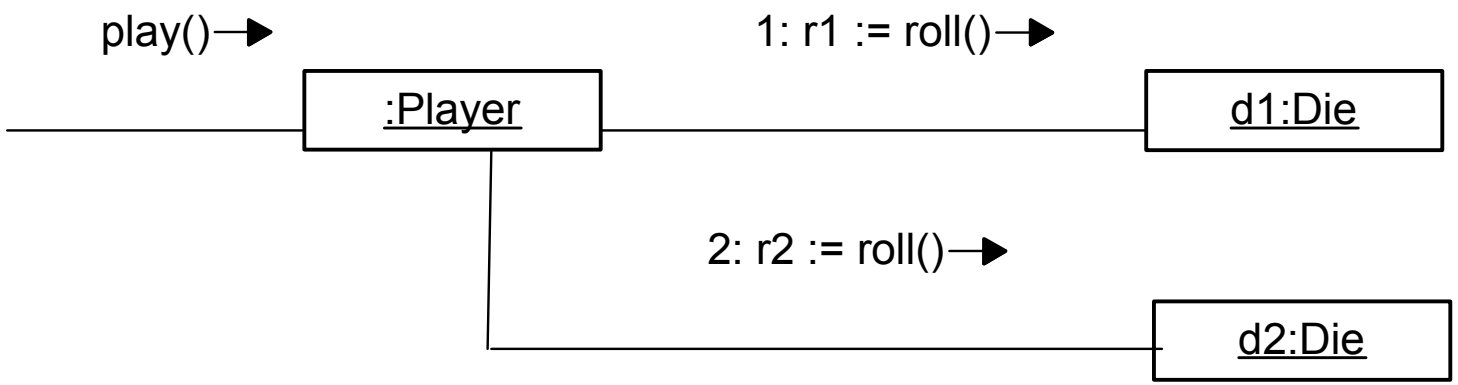
Conceptual model – מודל מושגים

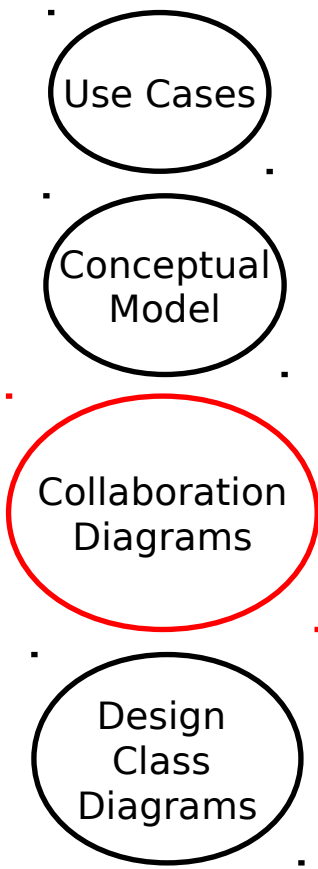




דוגמה – משחק קוביות

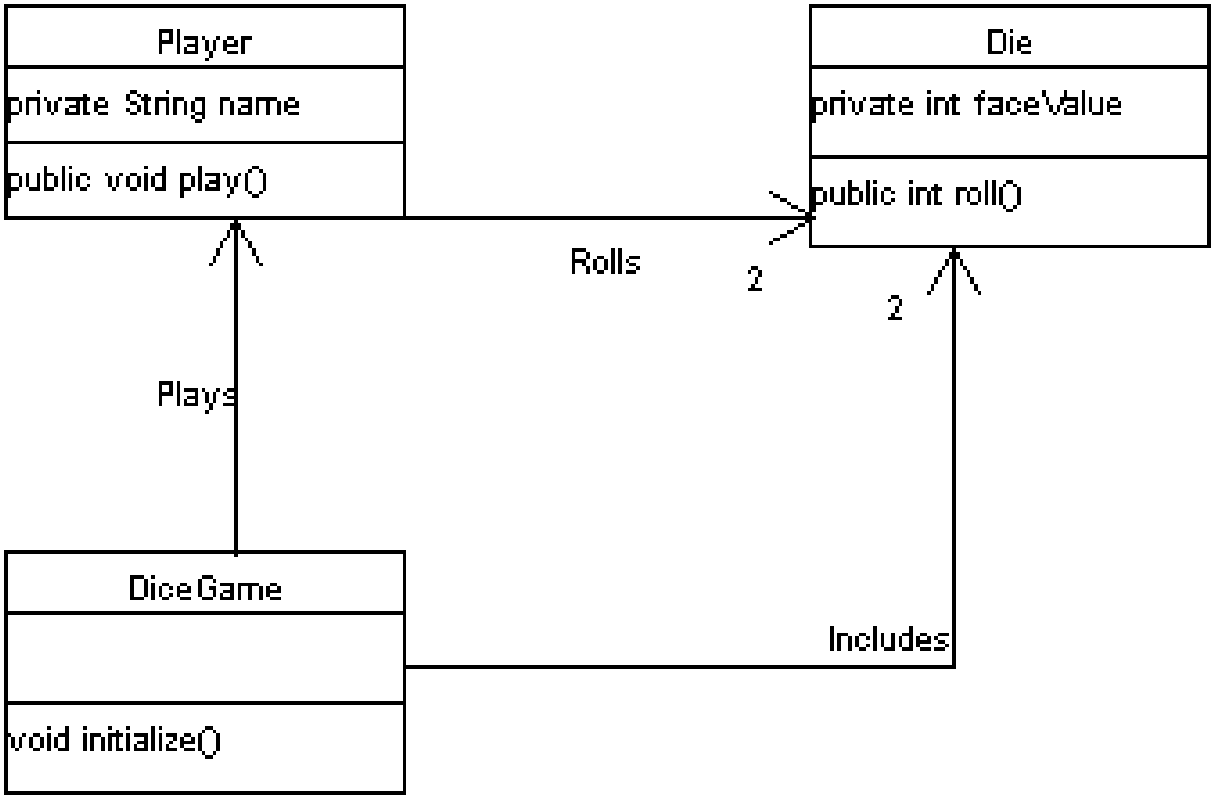
תרשים שיתוף-פעולה – Collaboration Diagram

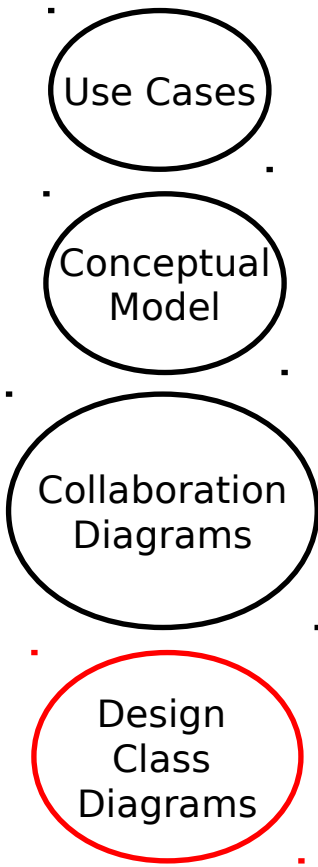




דוגמה – משחק קוביות

Design class Diagram – תרשים מחלקות





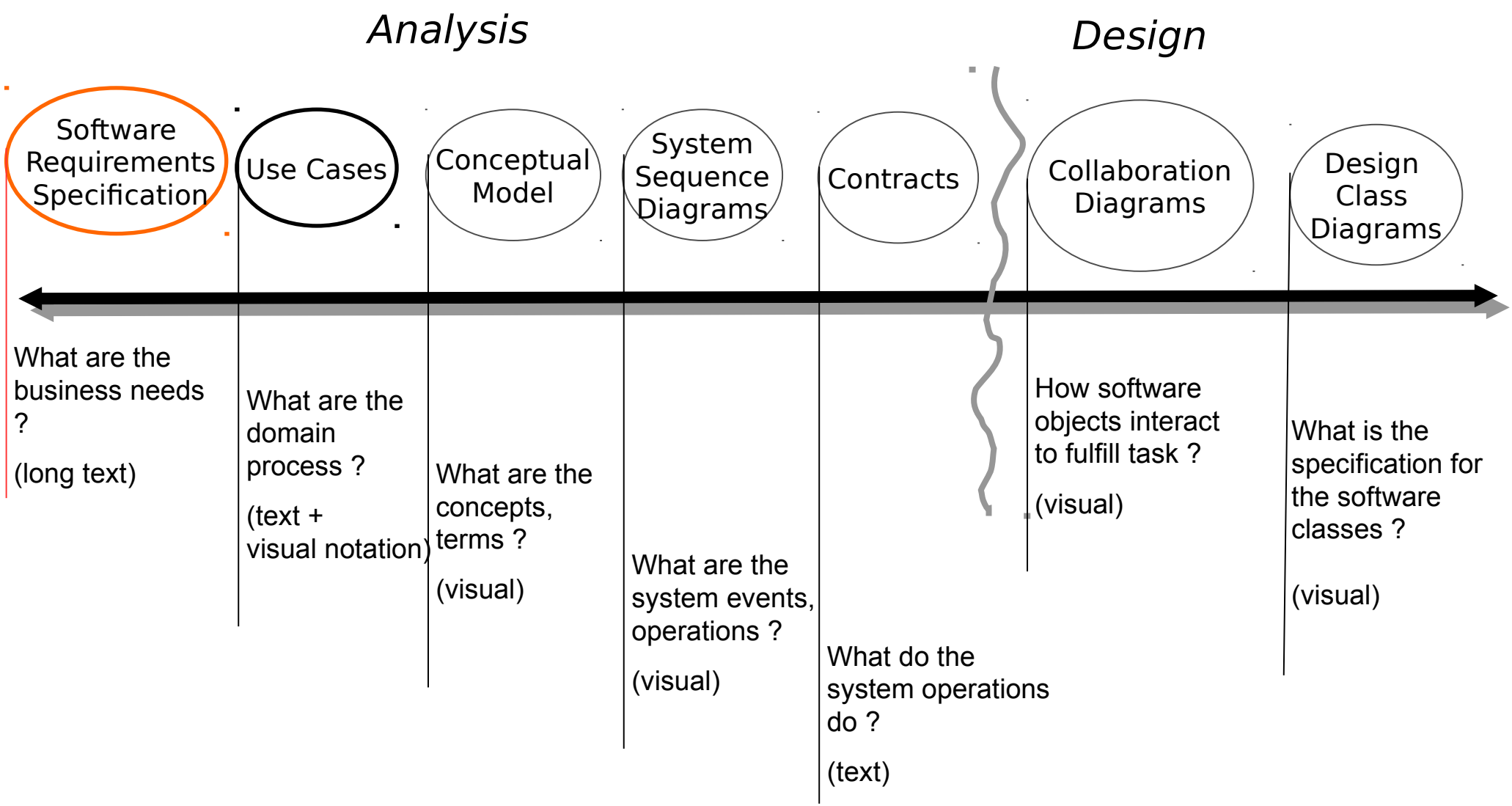
UML - Unified Modeling Language

- שפה מקובלת לכתיבת תרשימים של ניתוח ותיכנון.
- נועדה להקל על תקשורת בין לקוחות, מנהלים, מעצבים ומתכנתים לגבי מערכת-התוכנה.

דוגמה גדולה יותר: תוכנה לעמדת-מכירה

- עמדת-מכירה משמשת לרישום מכירות וקבלת תשלומים.
- כוללת חומרה ותוכנה.
- מתקשרת עם אפליקציות כגון של חברות-אשראי.

מפרט דרישות:
טקסט המתאר בפירוט מה המערכת צריכה לעשות



מפרט דרישות – פעולות

- R1.1 – Record the current sale – the items purchased
- R1.2 – Calculate current sale total, including tax and coupon calculations
- R1.3 – Capture purchased item information from a bar code or a manual entry of a universal product code (UPC)
- R1.4 – Reduce inventory quantities when sale is committed
- R1.5 – Log completed sale
- R1.6 – Cashier must login with an ID and password to use system
- R1.7 – Provide a persistent storage mechanism
- R1.8 – Provide inter-process and inter-system communication mechanism
- R1.9 – Display description and price of item recorded

מפרט דרישות – פעולות

- R2.1 – Handle cash payments, capturing amount tendered and calculating balance
- R2.2 – Handle credit card payments, capturing credit information from a card reader or by manual entry, and authorizing payment with store’s (external) credit authorization service via a modem connection
- R2.3 - Handle check payments, capturing ID card by manual entry, and by authorizing payment with the store’s (external) check authorization service via a modem connection
- R2.4 - Log credit payments to accounts receivable system, since the credit authorization service owes the store payment amount

מפרט דרישות – מאפיינים

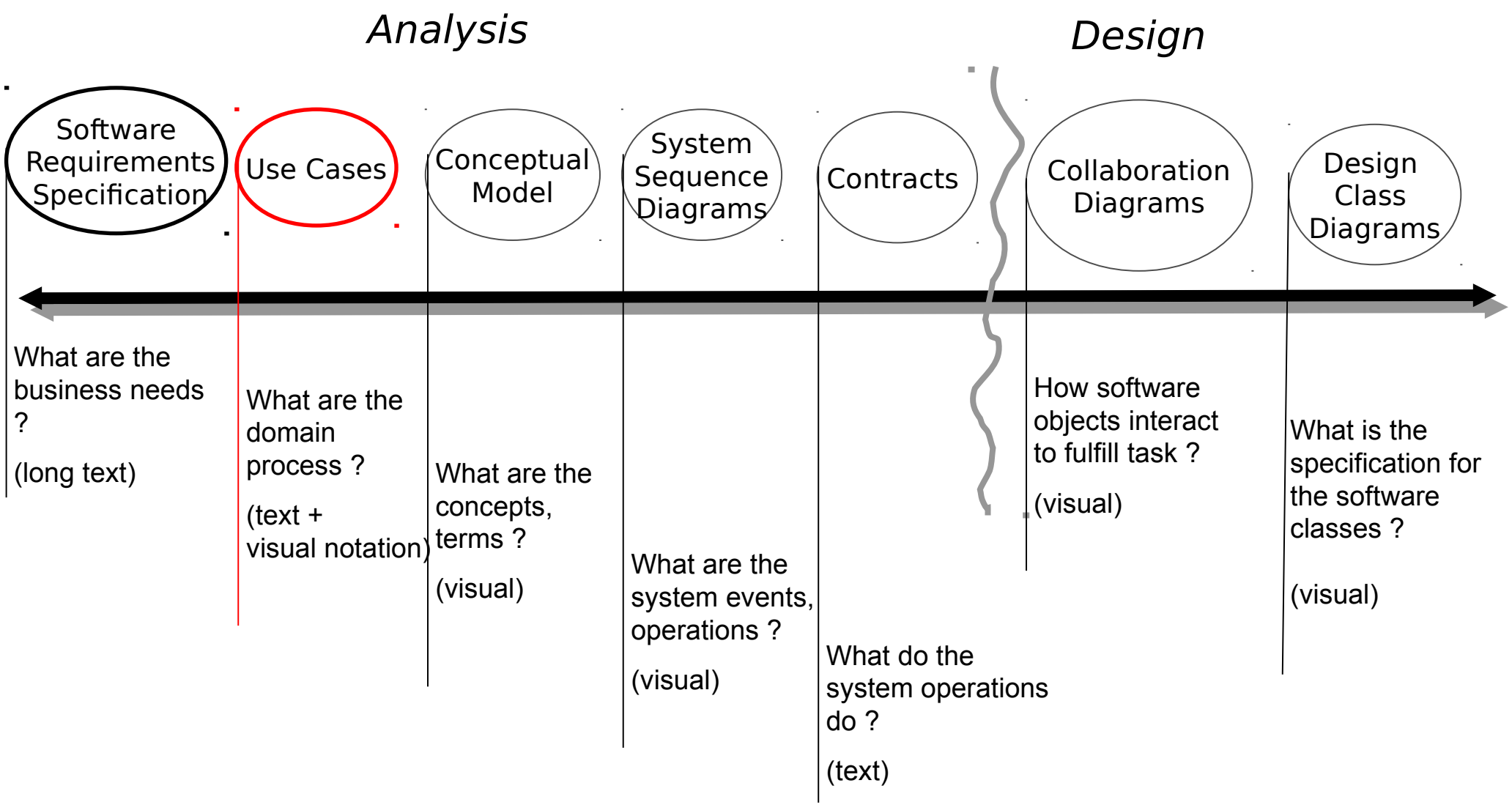
Attribute	Details and boundary constrains
Response time	When recording a sold item the description and price will appear within 5 seconds
Fault tolerance	Must log authorized credit payment to accounts receivable within 24 hr even if power fails or device failure
Operating system	Knoppix Linux

מפרט דרישות – קישור בין פעולות למאפיינים

Ref#	Function	Attribute	Details and Constrains
R1.1	Record the current sale – the items purchased		
R1.2	Calculate current sale total, including tax and coupon calculations		
R1.9	Display description and price of item recorded	Respons e time	5 seconds max.
R2.4	Log credit payments to accounts receivable system, since the credit authorization service owes the store payment amount	Fault tolerance	Must log to accounts receivable within 24 hr even if power fails or device failure

תרחישי שימוש:

"סיפורים" על שימוש במערכת, מנקודת מבט של שחקן חיצוני



תרחישי שימוש – למה צריך אותם?

- תקשורת - הסכמה על הדרישות מהמערכת.
- הצגת המערכת לצופה חיצוני (מנהל / משתמש).
- זיהוי אובייקטים שיהיו במערכת.
- בסיס לתוכנית בדיקות.
- בסיס למדריך למשתמש.
- קביעת סדרי עדיפויות.

תרחיש שימוש - דוגמה

Use Case: Buy Items

Actors: Customer, Cashier

Description: A *Customer* arrives at a checkout with items to purchase. *Cashier* records the purchased items and collects payment.

תרחיש שימוש – שלב א: סדר פעולות רגיל

Actor Actions:

1. This use case begins when a Customer arrives at a POST checkout system with items to purchase

2. Cashier records the identifier for each item. If there is more than one of the same item, Cashier enters the quantity

4. On completion of item entry, Cashier indicates to POST that item entry is complete

6. Cashier tells Customer the total

System Response:

3. Determines item price and adds item info to the running sales transaction
   Description and price of the current item is presented

5. Calculates and presents sale total

7. Customer gives a cash payment

8. Cashier records the cash received

9. Shows the balance due back to Customer Generates a receipt

10. Cashier deposits the cash received and extracts balance owing
Cashier gives balance owing and receipt to Customer

11. Logs the completed sale

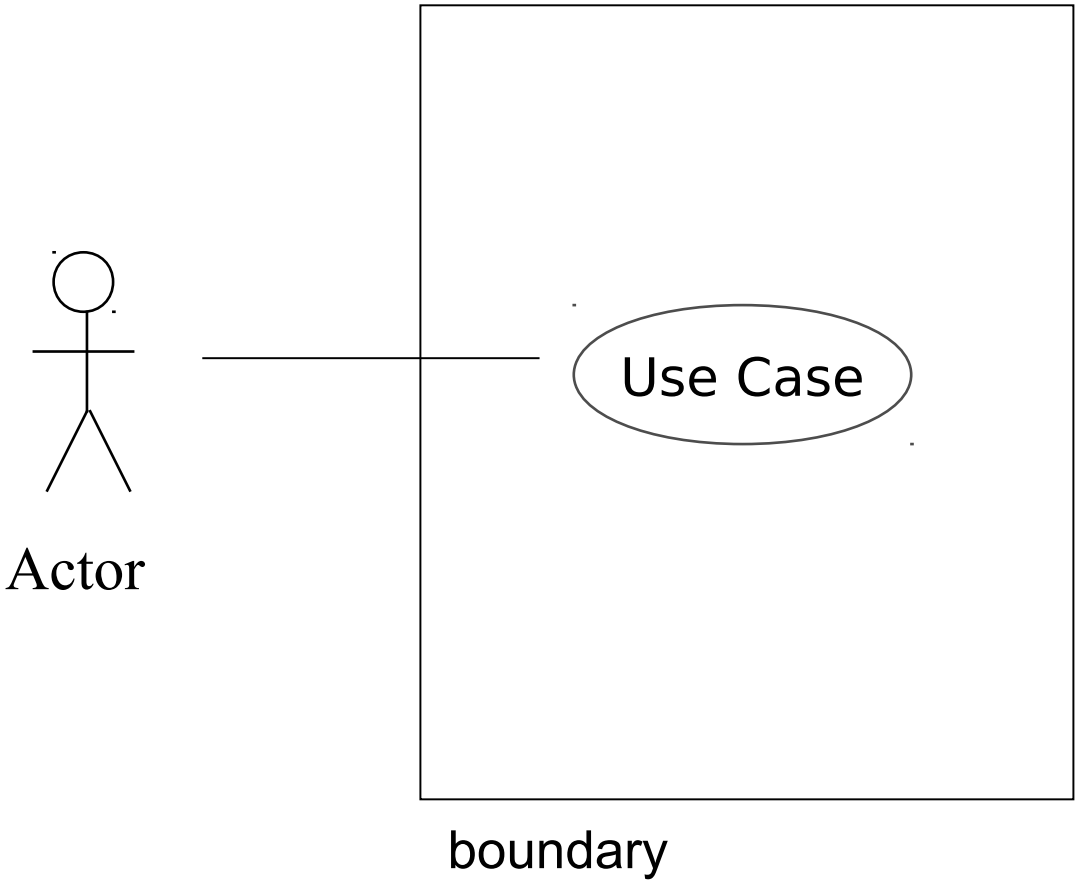
12. Customer leaves with items purchased

תרחיש שימוש – שלב ב: נתיבים חלופיים

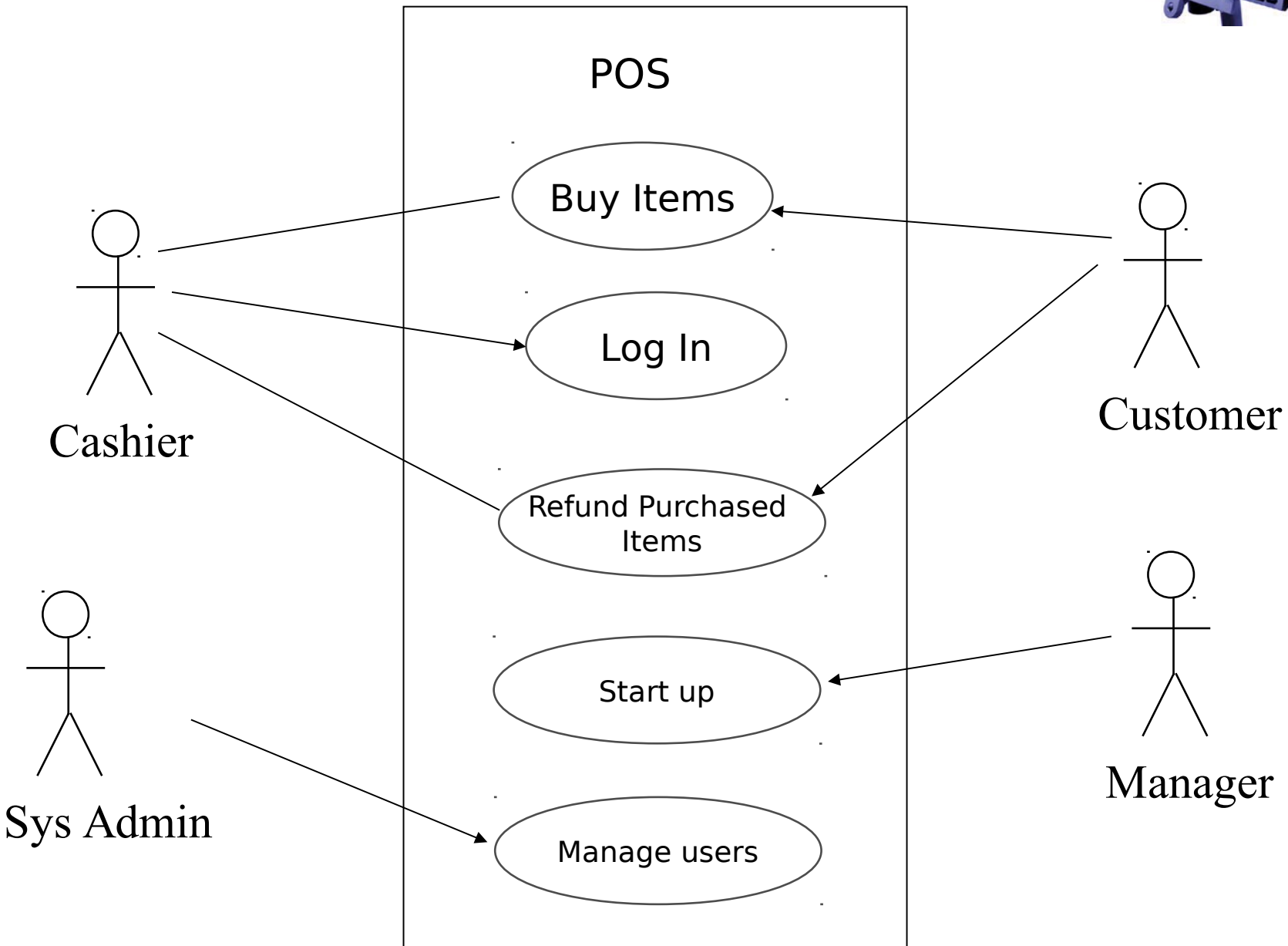
Line 2: Invalid identifier entered. Indicate error

Line 7: Customer did not have enough cash.
Cancel sales transaction

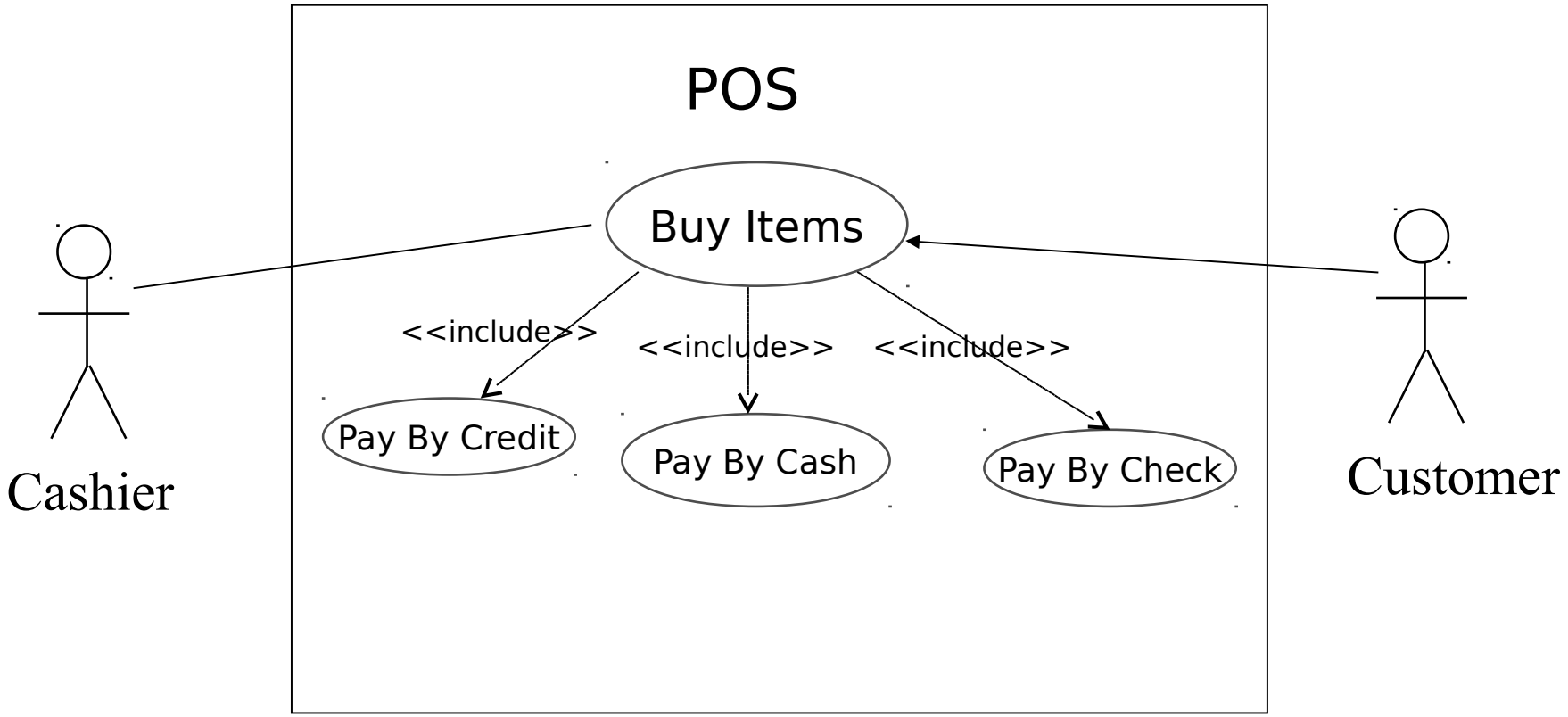
רישום תרחישי-שימוש ב-UML



רישום תרחישי-שימוש ב-UML



קשרים בין תרחישי-שימוש



כלים אוטומטיים

- יש כלים אוטומטיים ליצירת תרשימים ב-UML.
- **דוגמה**: פאפירוס, ניתן להתקין באקליפס:
  https://www.eclipse.org/papyrus/download.html
- **לא חייבים להשתמש בכלים או ב UML** – אפשר לכתוב תרחישי-שימוש כמסמכי-טקסט.
- **העיקר שתהיה תקשורת טובה** בין כל המעורבים בפרוייקט – לקוחות, מנהלים ומתכנתים.

תרחישי שימוש - דגשים

- תרחישי-שימוש נבנים יחד עם הלקוחות ובשפה שהם מבינים.
- מתחילים מתרחישים כלליים, ואחר-כך יורדים לפרטים לפי סדרי-העדיפויות של הלקוחות.
- תרחישי שימוש **לא** כוללים פירוט של אלגוריתמים, מבני-נתונים, ופקדים בממשק-משתמש.

תרחישי שימוש – דוגמה נוספת

אתר אינטרנט להזמנת ארוחות:

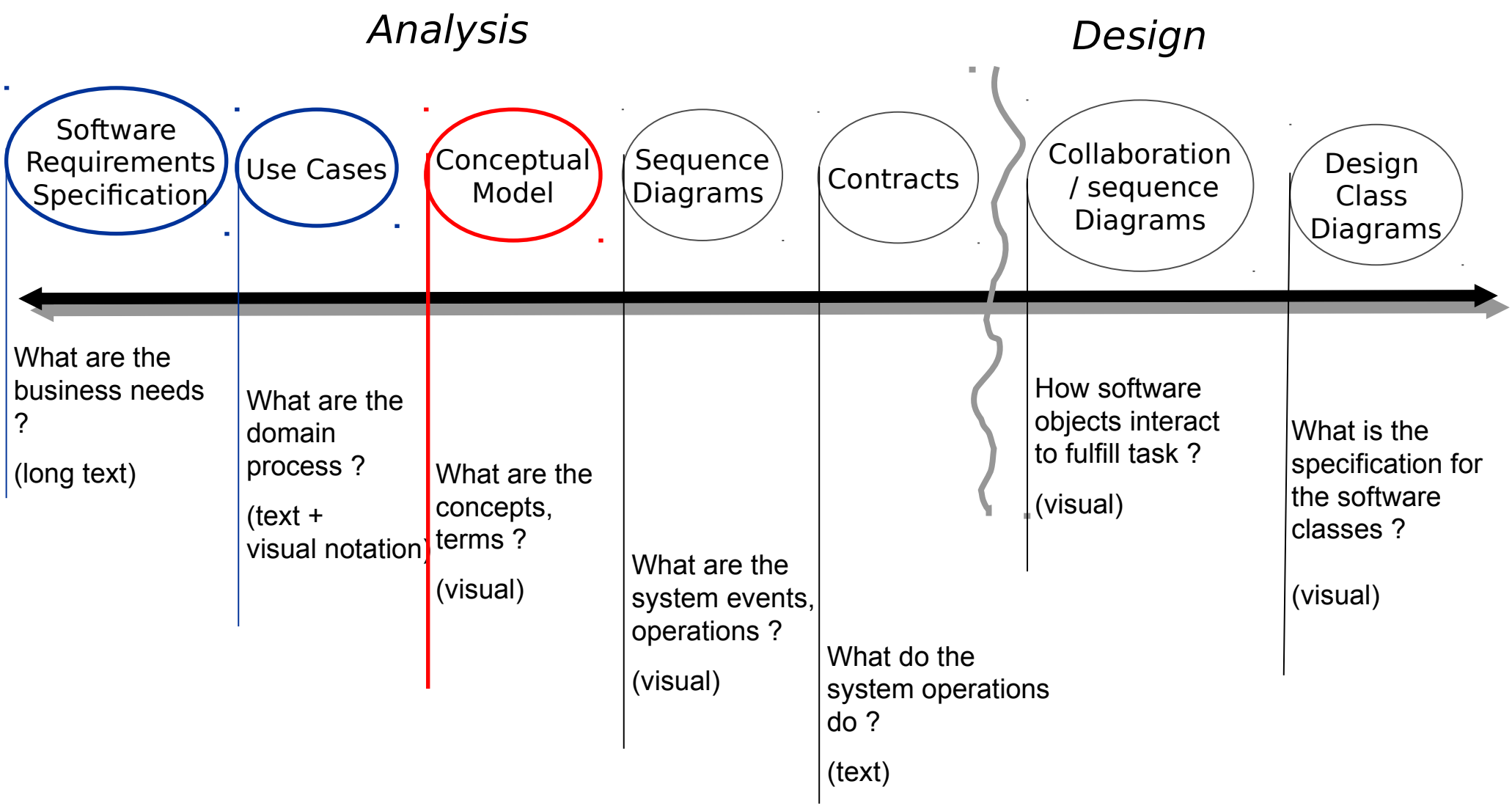
1)מטרת האתר היא לאפשר ללקוחות להזמין ארוחות אונליין.

2)לקוח צריך להיכנס למערכת עם שם וסיסמה.

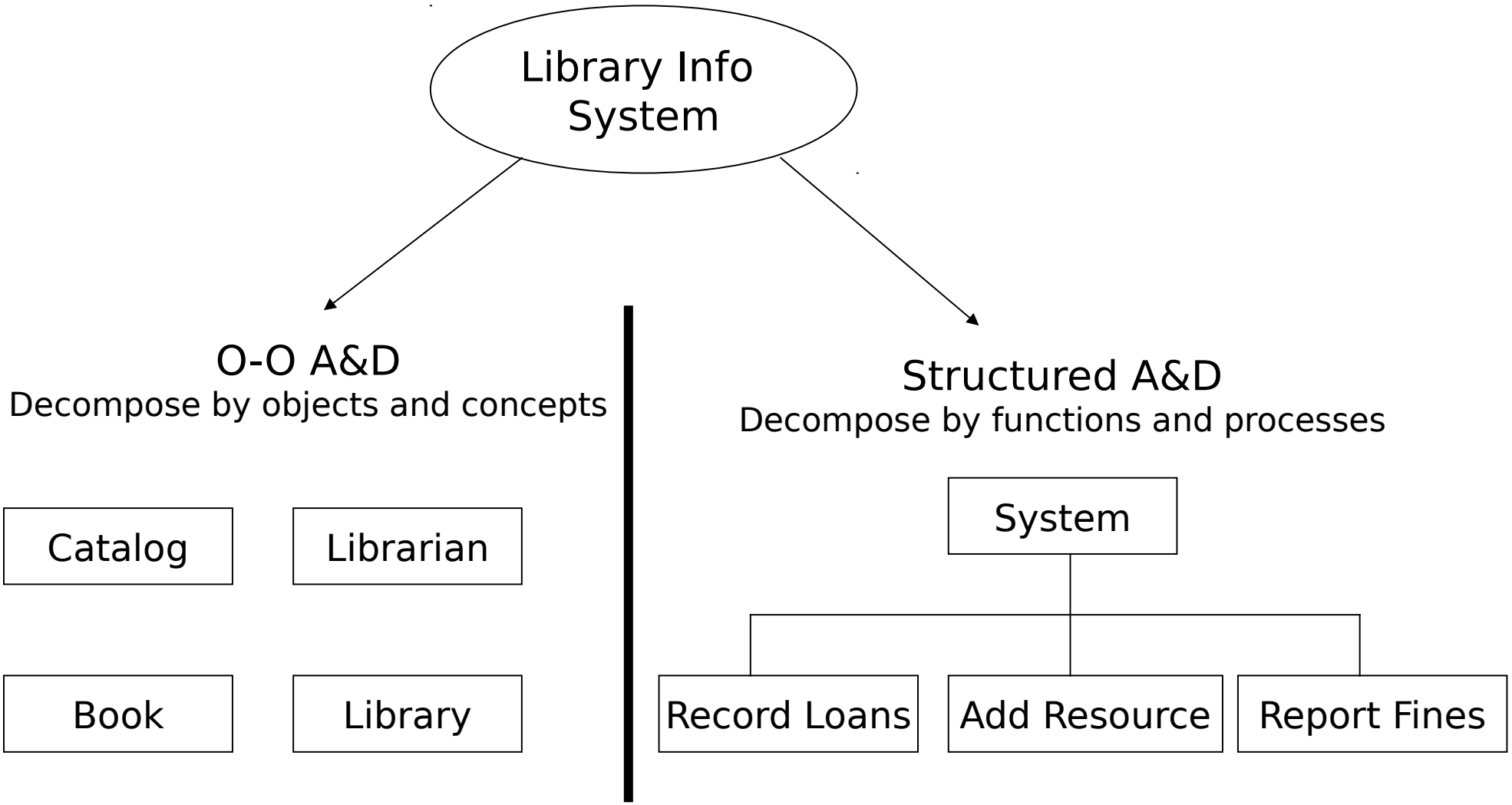
3)אחרי הכניסה למערכת הלקוח יכול להזמין ארוחה ואז לשלם עליה בכרטיס אשראי או בפייפאל.

תרחישי שימוש – דוגמה בפאפירוס

מודל מושגים:
תרשים של מושגים ומערכות שהתוכנה עובדת איתם, והקשרים ביניהם



תיכנון מונחה-עצמים
לעומת
תיכנון מונחה-פונקציות



איך מזהים את המושגים הרלבנטיים?

--- מתוך שמות-העצם בתרחישי השימוש:

Actor Actions:

1. This use case begins when a **Customer** arrives at a **POST checkout** system with **items** to purchase

2. **Cashier** records the **identifier (UPC)** for each item.
   If there is more than one of the same **item**, **Cashier** enters the **quantity**

System Response:

3. **Determines item price** and adds item to the running **sale transaction**
   Description and **price** of the current **item** is presented

מושגים הקשורים לנקודת-מכירה

What are the concepts in the
Point of Sale system ?

POST	Item	Store	Sale
SalesLineItem	Cashier	Customer	Manager
Payment	ProductCatalog	ProductSpecification	

מודל מושגים - **קשרים**

סוגים נפוצים של קשרים

Drawer - POST	**A is physical part of B**
SalesLineItem – Sale	**A is a logical part of B**
POST – Store, Item – Shelf	**A is physically contained in/on B**
ItemDescription – Item	**A is logically contained in B**
SalesLineItem – Sale	A is a line item of a transaction or report B
Sale – POST	**A is known/ logged/ recorded/ reported/ captured in B**
Cashier – Store	A is a member of B

***High priority associations**

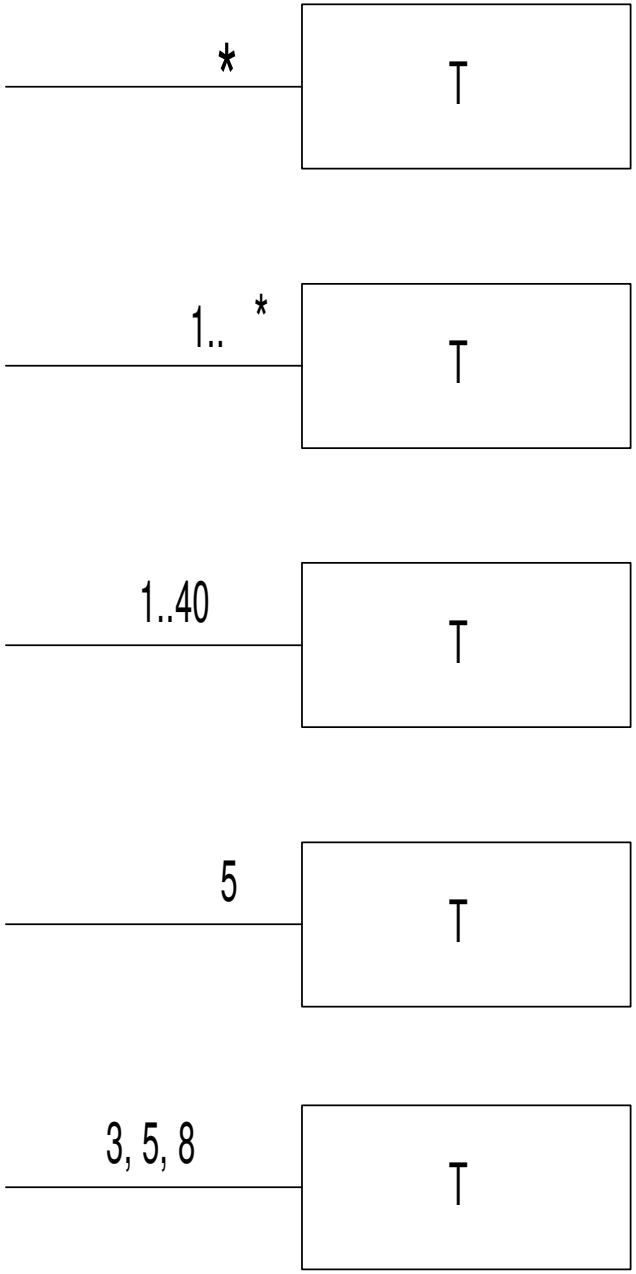
סוגים נפוצים של קשרים

A uses or manages B	*Cashier – POST*
A communicated with B	*Customer – Cashier*
A is related to a transaction B	*Customer – Payment*
A is transaction related to another transaction B	*Payment – Sale*
A is next to B	*POST – POST*
A is owned by B	*POST - Store*
A is an organizational subunit of B	*Department - Store*

קשרים - ריבוי

- אפשר לציין ליד כל קצה של הקשר, כמה עצמים מסוג א קשורים לכל עצם מסוג ב.
- דוגמה: "בחנות אחת יכולים להיות הרבה עצמים":

קשרים - ריבוי



zero or more;
"many"

one or more

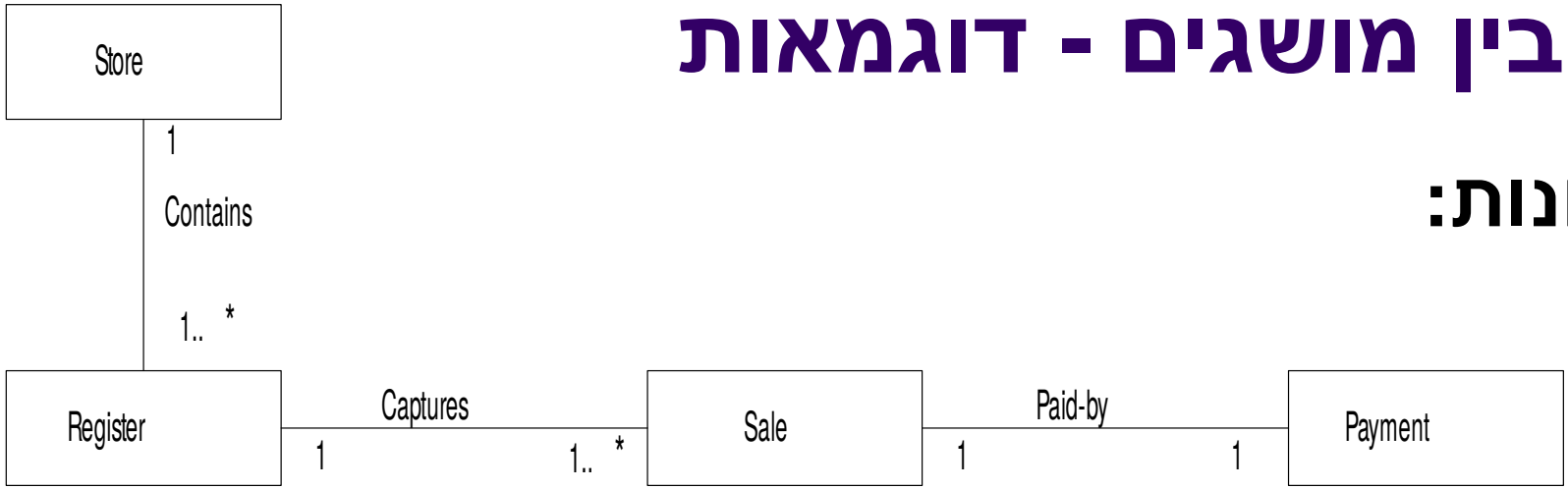
one to 40

exactly 5

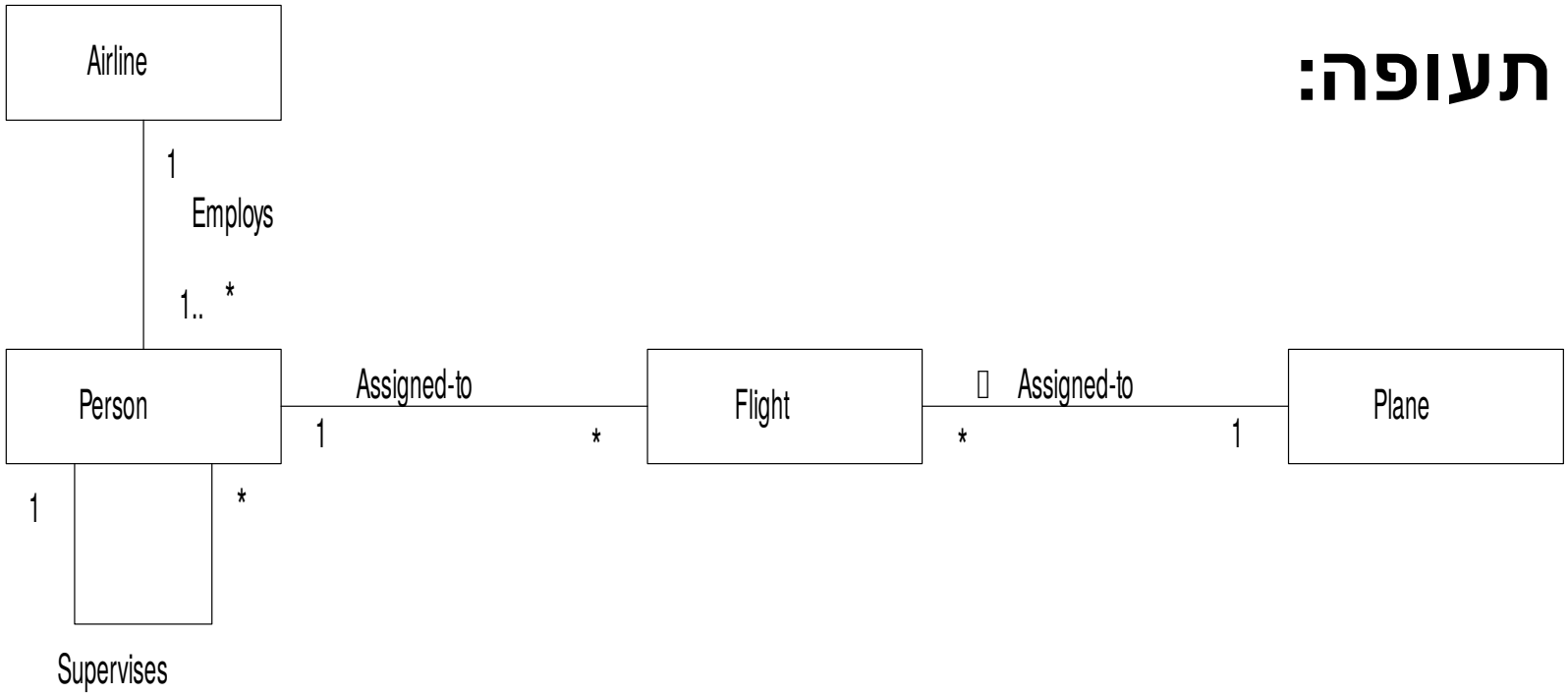
exactly 3, 5, or 8

קשרים בין מושגים - דוגמאות

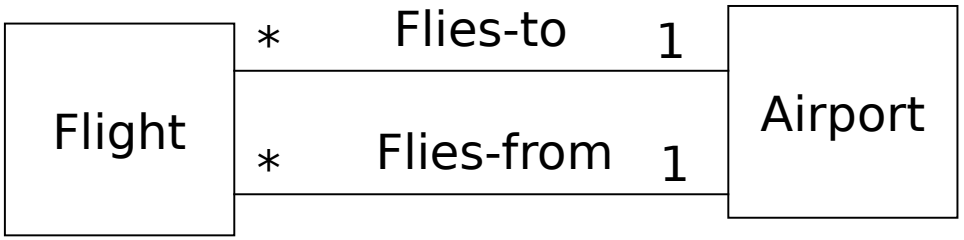
חנות:

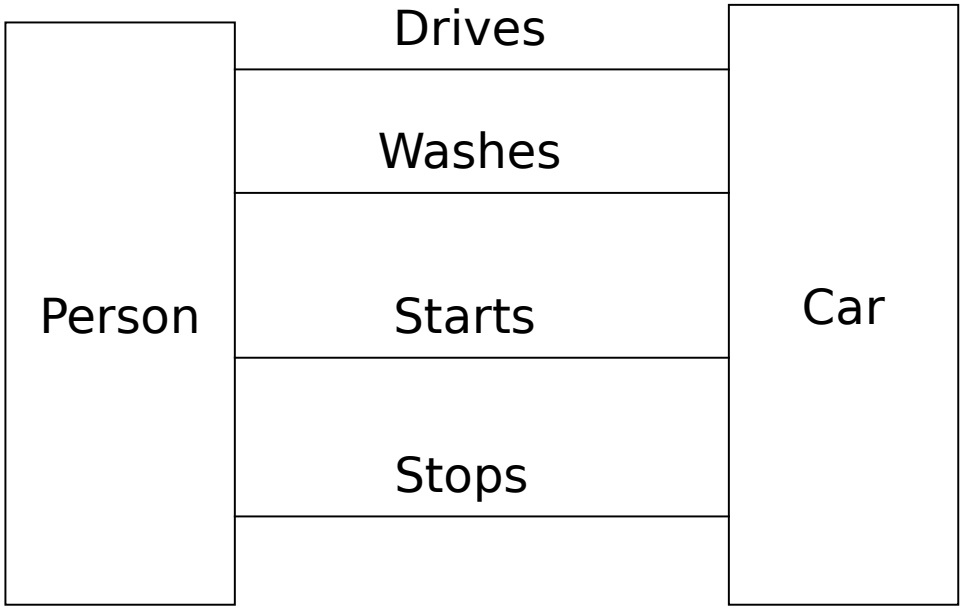


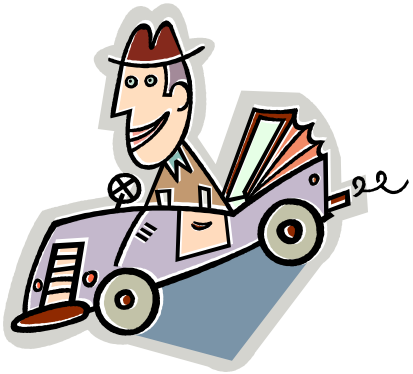
חברת תעופה:



קשרים מרובים

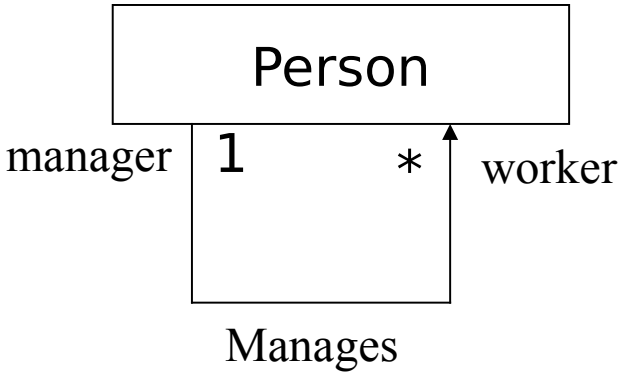


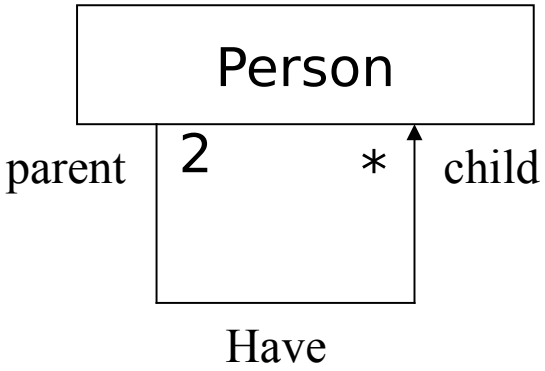




קשרים רקורסיביים

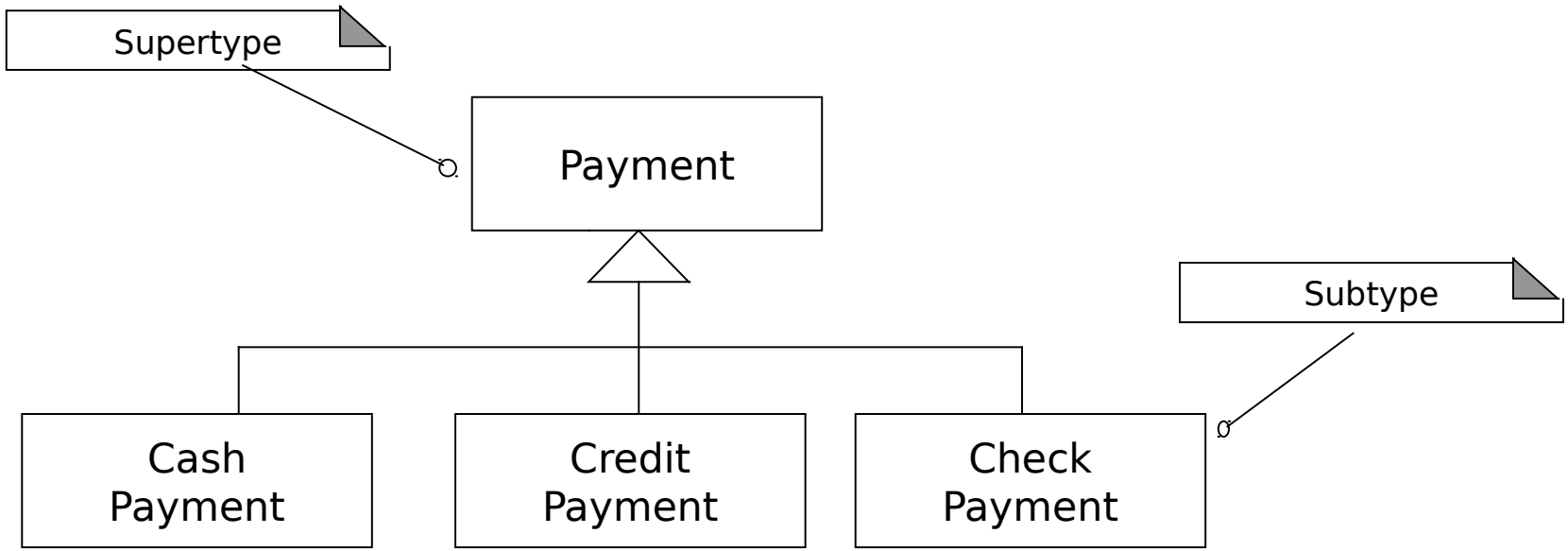
קשר בין מושג לעצמו:





קשרי כלל ופרט ("ירושה"):

מושג ב הוא *סוג של* מושג א:



מתי להגדיר קשרי כלל ופרט?

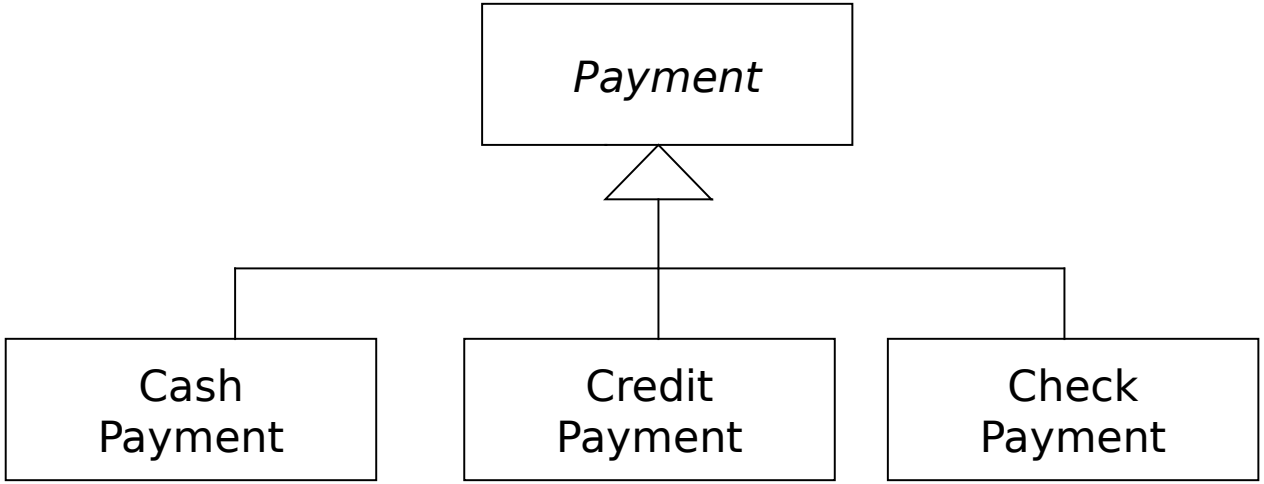
- כשלסוג הפרטי יש *מאפיינים* נוספים:
  - Library*: Book*, a subtype of *Loanable* resource, has ISBN attribute

כשלסוג הפרטי יש *קשרים* נוספים:

  - POST: *CreditPayment*, a subtype of *Payment*, is associated with a *CreditCard*

מושגים מופשטים

- אם כל עצם ששייך למושג כללי כלשהו, חייב להיות שייך לתת-מושג כלשהו שלו – אז המושג הכללי נקרא **מופשט** (abstract).
- **דוגמה**: אם כל תשלום חייב להיות תשלום במזומן, תשלום באשראי או תשלום בצ'ק, אז "תשלום" הוא מושג מופשט.



בניית מודל מושגים עבור נקודת-מכירה

ProductCatalog

ProductSpecificatin

SalesLineItem

Store

Item

Sale

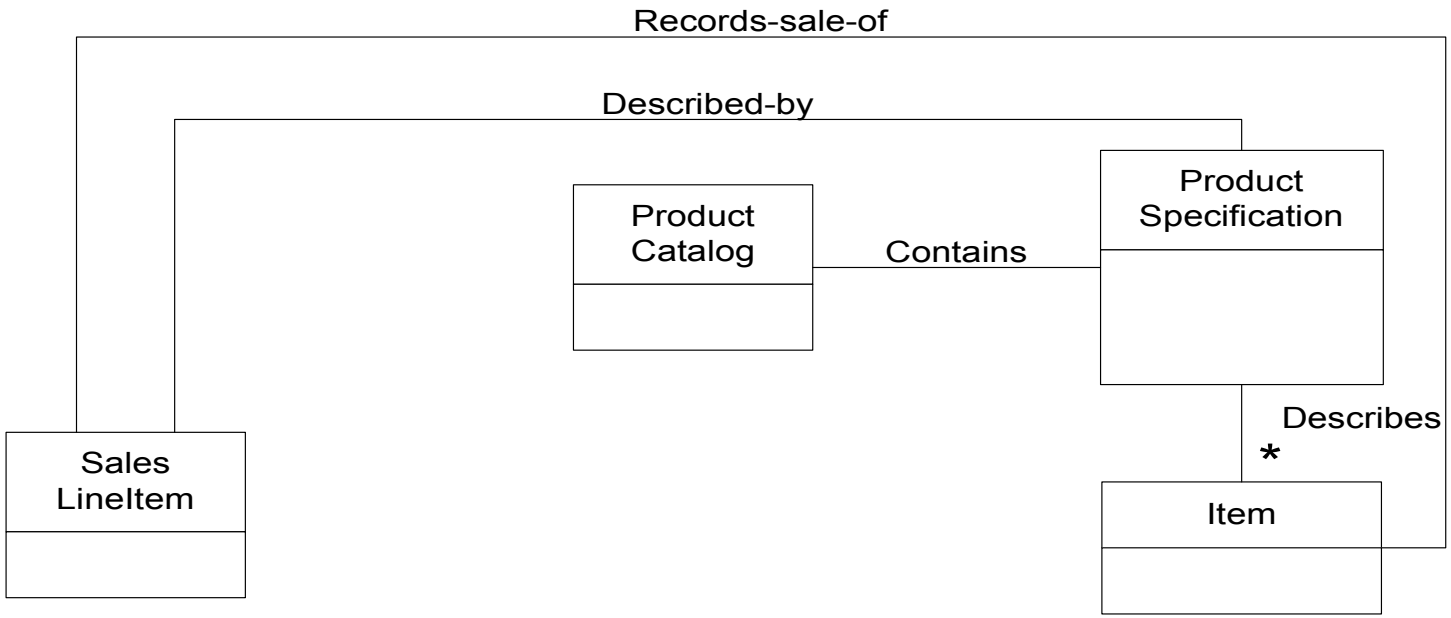
Register

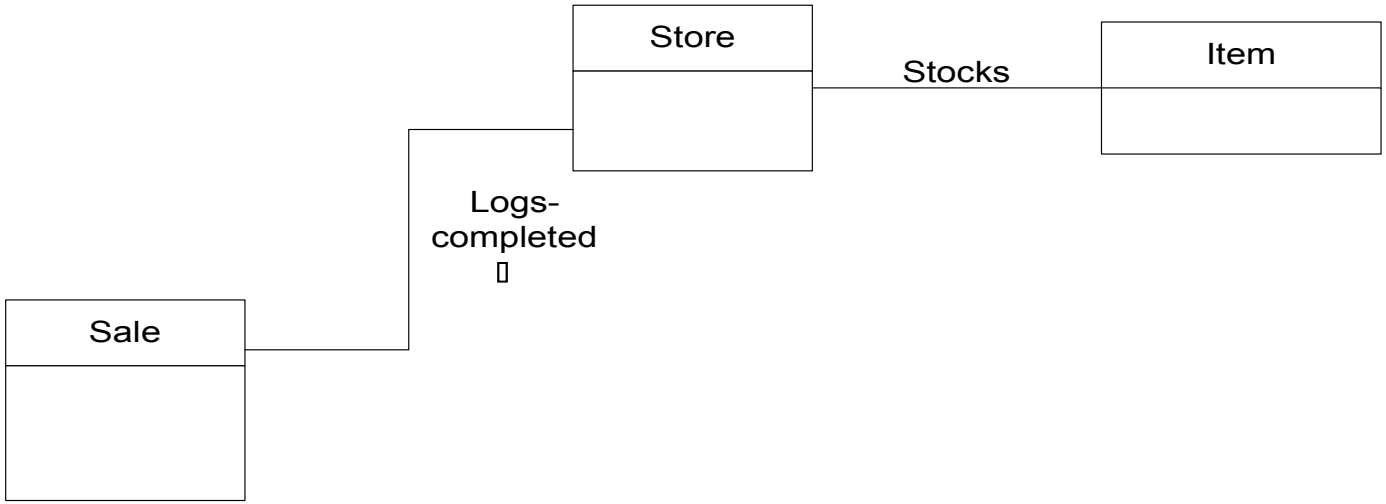
Manager

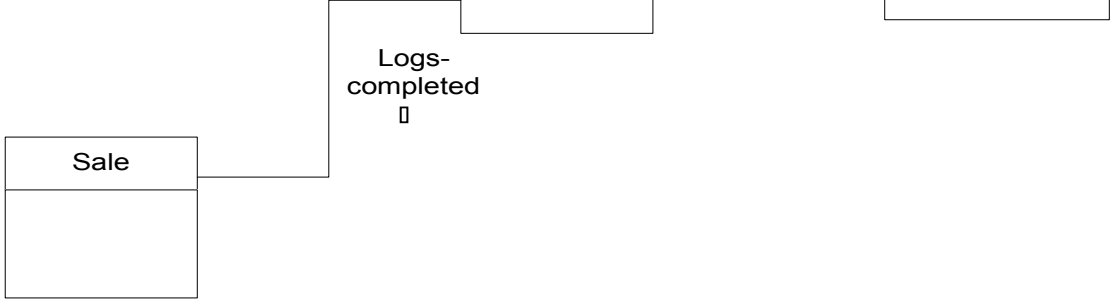
Payment

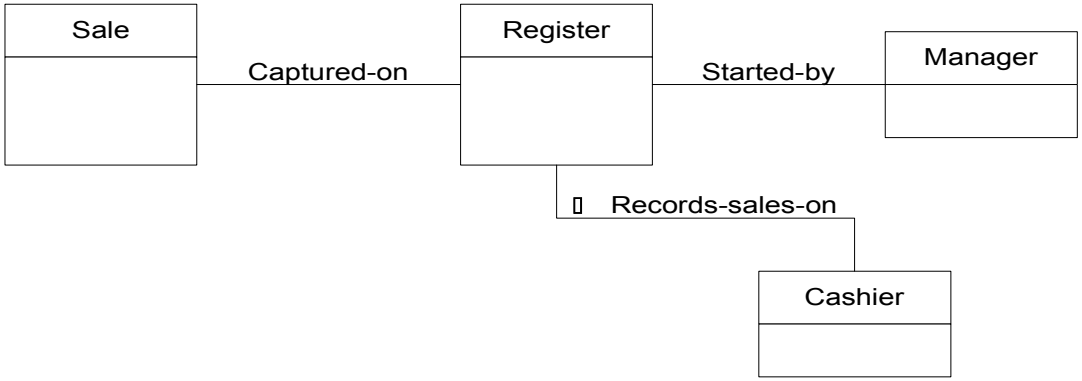
Customer

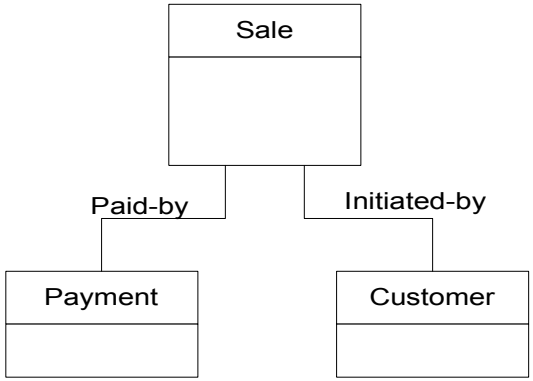
Cashier

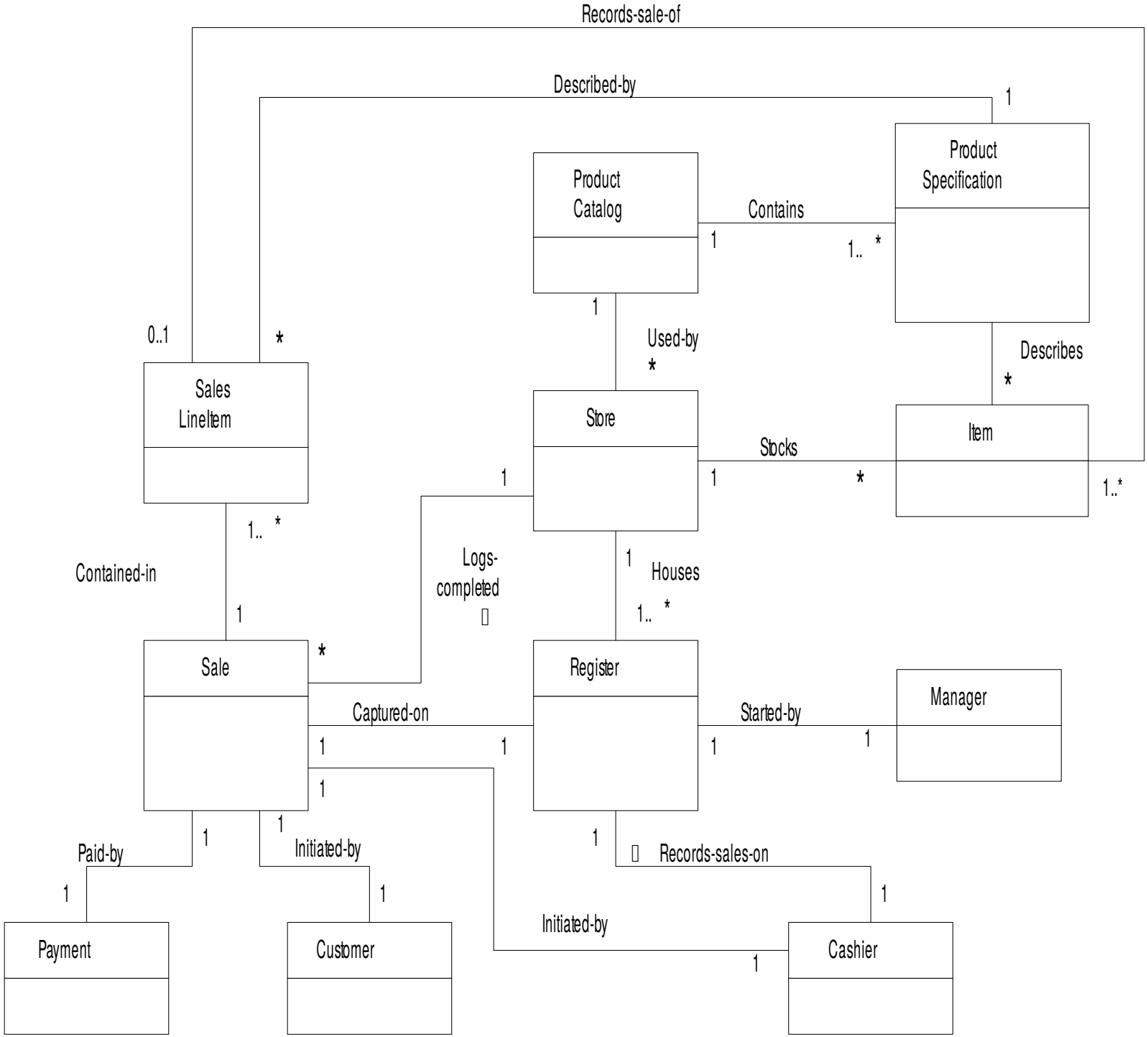

Records-sale-of
Described-by
Product Catalog
Contains
Product Specification
Describes
*
Sales LineItem
Item


Store
Stocks
Item
Logs-completed
Sale


Logs-
completed
Sale

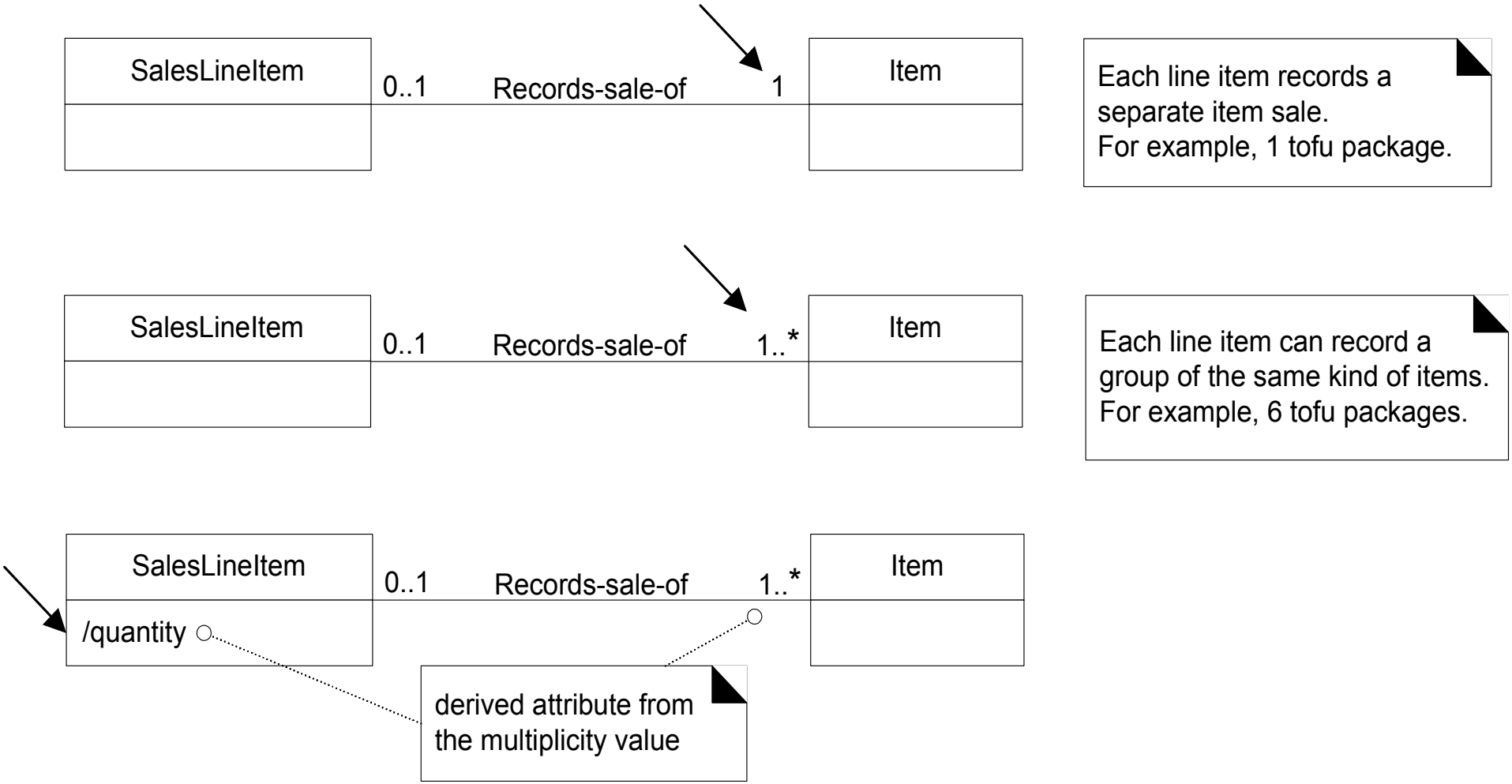

Sale
Captured-on
Register
Started-by
Manager
Records-sales-on
Cashier


Sale
Paid-by
Initiated-by
Payment
Customer


Records-sale-of
Described-by
1
Product Specification
Product Catalog
Contains
1
1..*
1
Used-by
*
Describes
*
0..1
*
Sales LineItem
Store
Item
Stocks
1
1
*
1..*
1..*
Contained-in
Logs-completed
1
Houses
1..*
1
Sale
*
Register
Manager
Captured-on
Started-by
1
1
1
1
1
1
1
Paid-by
Initiated-by
1
Records-sales-on
1
1
1
Initiated-by
Payment
Customer
Cashier
1

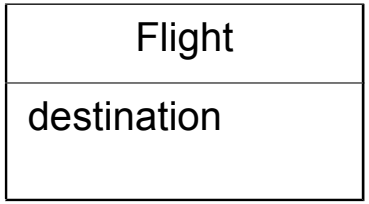
מודל מושגים - **מאפיינים**

זיהוי מאפיינים מתוך תרשים המושגים



מאפיינים צריכים להיות **פשוטים**.
אם הם מורכבים – הם כנראה **מושגים**.

Worse



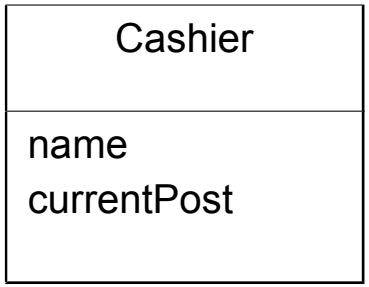
destination is a complex concept

Better

Relate concepts with an association, not an attribute

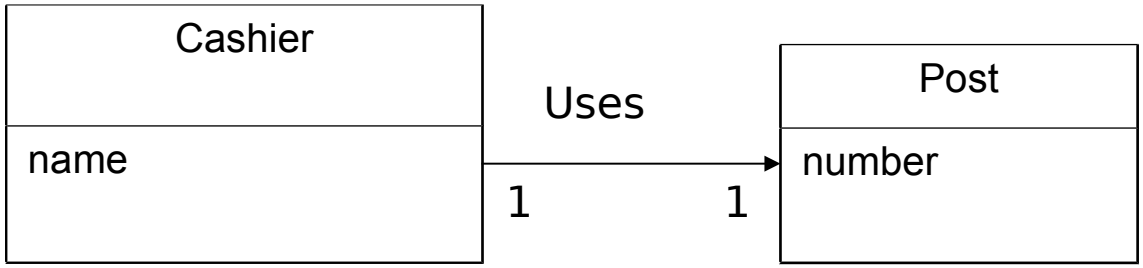
מאפיינים לעומת מושגים – דוגמה נוספת

Worse



currentPost is not a simple attribute

Better



Relate concepts with an association, not an attribute

מאפיינים לעומת מושגים – דוגמה - Item

- *Item* instance represents a physical item in a store
- *Item* has a description, price and UPC which are not recorded anywhere else

Item
Description Price SerialNumber UPC

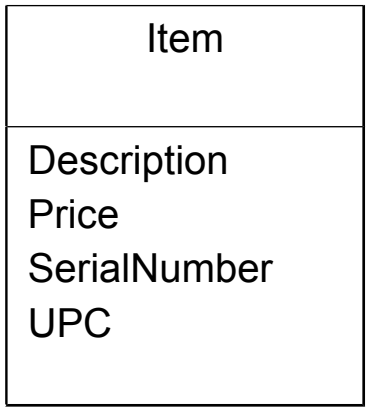
מאפיינים לעומת מושגים – Item - בעיות

- Duplicate data (description, price, UPC)
  - Very space inefficient
- Assume store sells out *Item X*,
  - Can we answer a question like:
    "how much *item X* cost"?

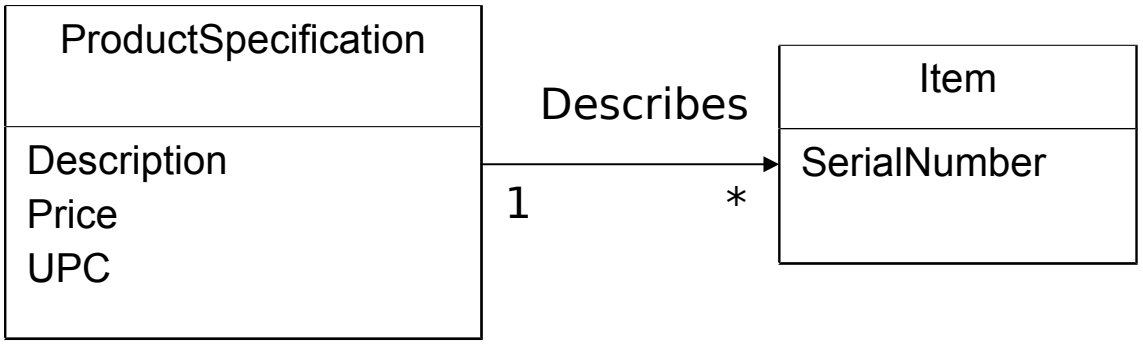
מאפיינים לעומת מושגים – Item - בעיות

- Solution to *Item* concept problem is to add a new concept called ***ProductSpecification***
  - It represents a description of information about items

Worse



Better

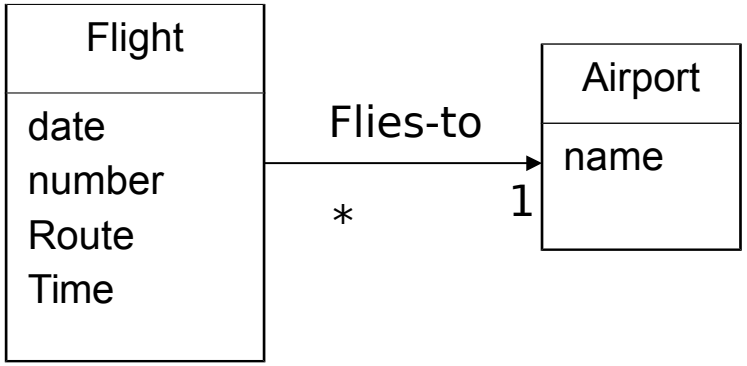


מאפיינים לעומת מושגים – דוגמה - Item

- *Item* instance represents a physical item in a store
- *Item* has a description, price and UPC which are not recorded anywhere else

Item
Description Price SerialNumber UPC

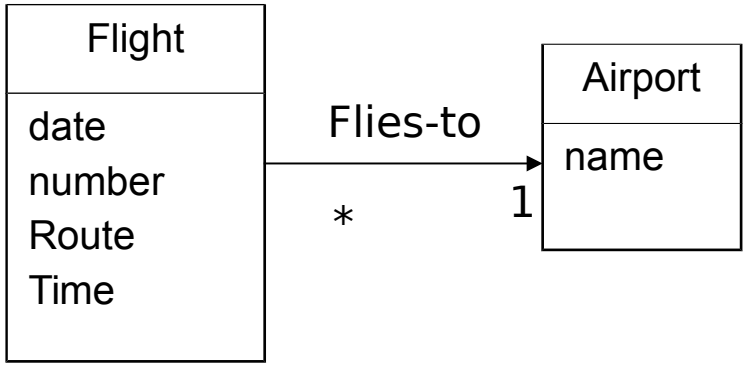
מאפיינים לעומת מושגים – דוגמה - טיסה



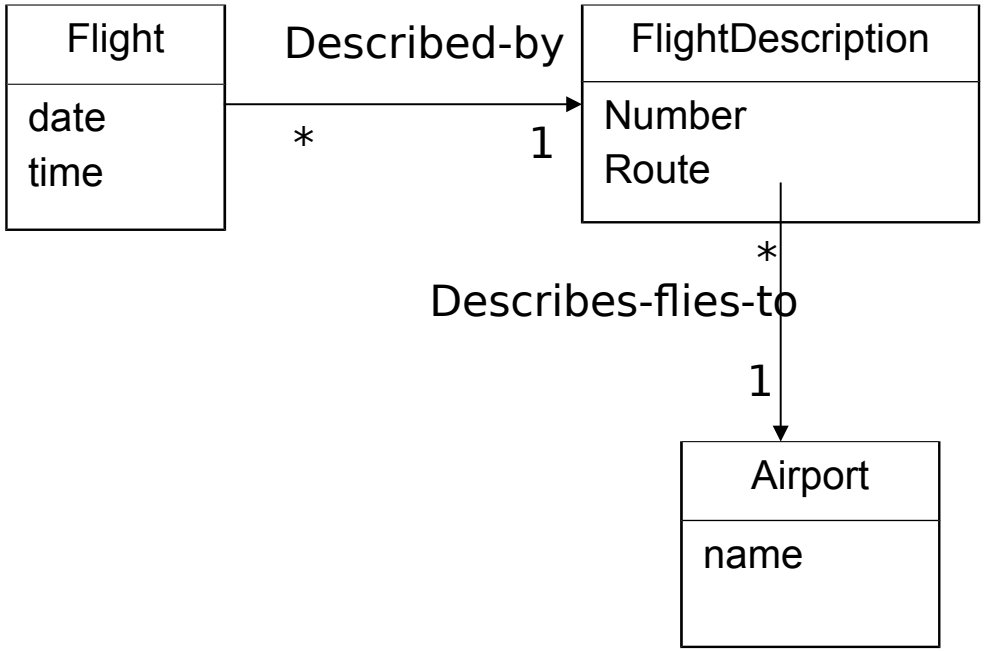
- All the flights are cancelled for 6 months
  - How can you get record of flight routes?

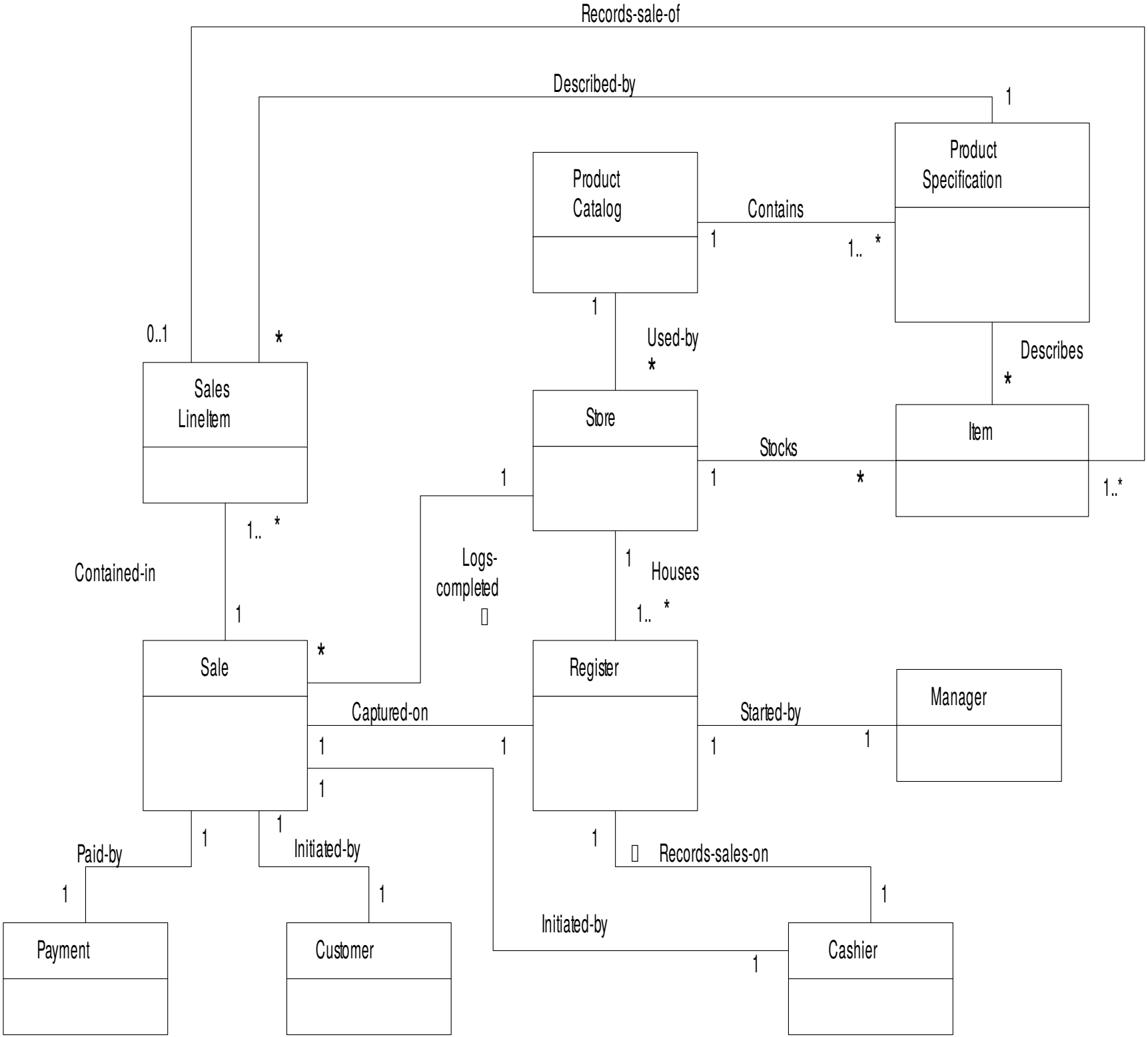
מאפיינים לעומת מושגים – דוגמה - טיסה

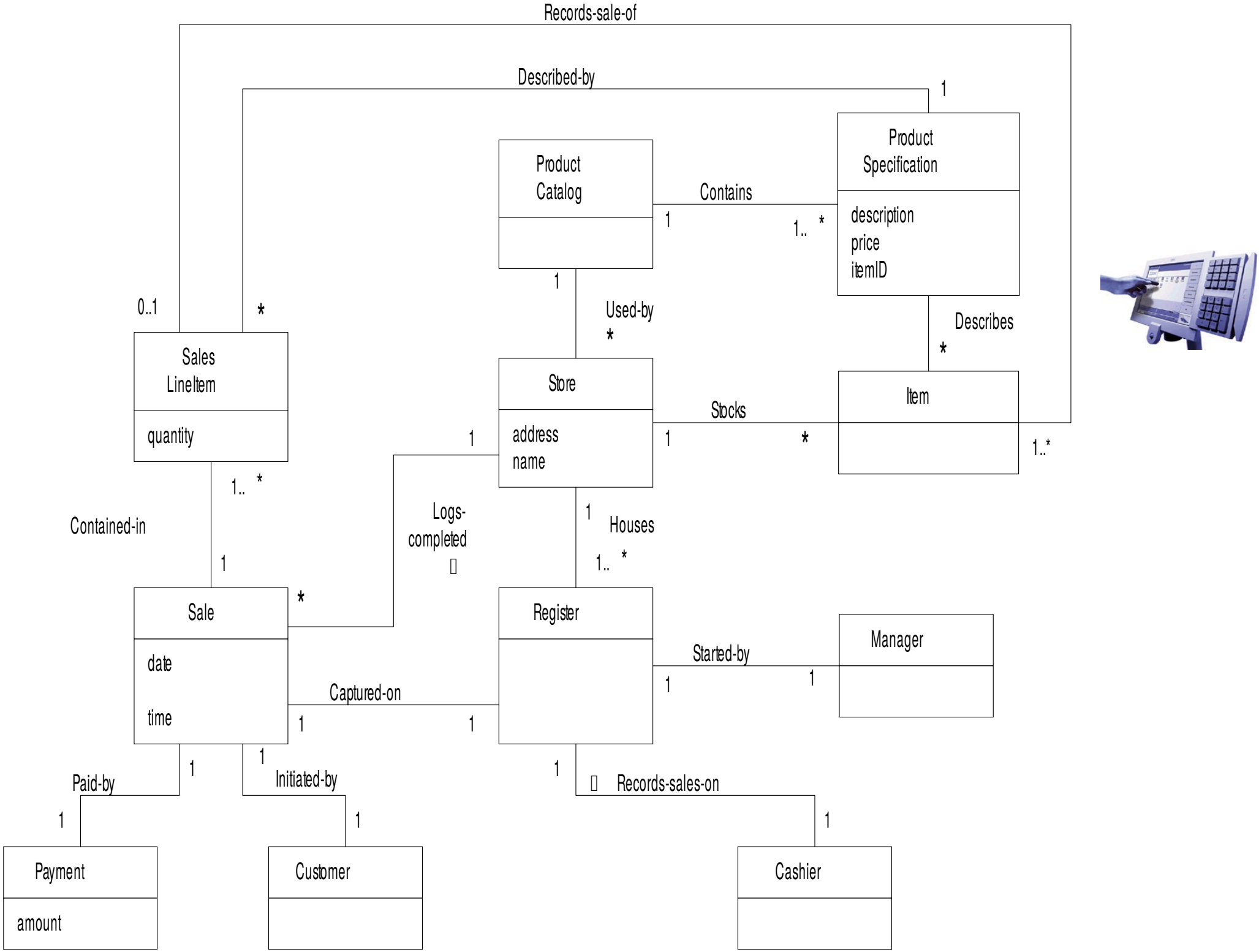
Worse



Better




Records-sale-of
Described-by
1
Product Specification
Product Catalog
Contains
1
1..*
1
Used-by
*
Describes
*
0..1
*
Sales LineItem
Store
Item
Stocks
1
1
*
1..*
1..*
Contained-in
Logs-completed
1
Houses
1..*
1
Sale
*
Register
Manager
Captured-on
Started-by
1
1
1
1
1
1
1
Paid-by
Initiated-by
Records-sales-on
1
1
1
Initiated-by
1
Payment
Customer
Cashier


Records-sale-of
Described-by
1
Product Specification
description
price
itemID
Product Catalog
Contains
1
1..*
1
Used-by
*
Describes
*
0..1
*
Sales LineItem
quantity
Store
address
name
Stocks
1
*
1..*
Item
1..*
Contained-in
1
1
Logs-completed
1
Houses
1..*
Sale
date
time
*
Register
Manager
Started-by
1
1
Captured-on
1
1
1
1
Paid-by
1
Initiated-by
1
1
Records-sales-on
1
Payment
amount
Customer
Cashier

“Royal & Loyal” system

- חברת R&L מנהלת "תוכניות נאמנות" עבור חברות המציעות ללקוחותיהם סוגים שונים של תגמולים. ("יש כרטיס מועדון ?...)

- התגמולים יכולים להתבטא בנקודות בונוס (ויזה...), קילומטרז' נצבר ("הנוסע המתמיד") או בונוסים אחרים כגון: תעריפים מועדפים, הגדלת סוג הרכב המושכר, שירות מועדף וכו'.

- כל סוג שירות שחברה מציעה ללקוחותיה יכול להיות מבוטא במערכת.

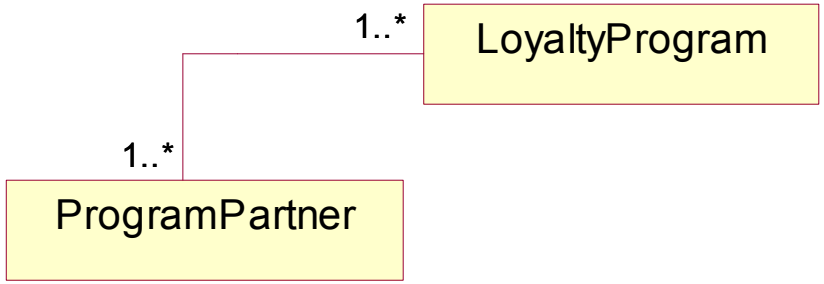
“Royal & Loyal” system

- ציטוט : " חברת R&L מנהלת "תוכניות נאמנות" "...
- "תוכנית נאמנות" – *LoyaltyProgram* - היא המשאב המרכזי אותו מנהלת החברה.
- חברת R&L צריכה לנהל אוסף של תוכניות נאמנות לכל לקוחותיה.

LoyaltyProgram

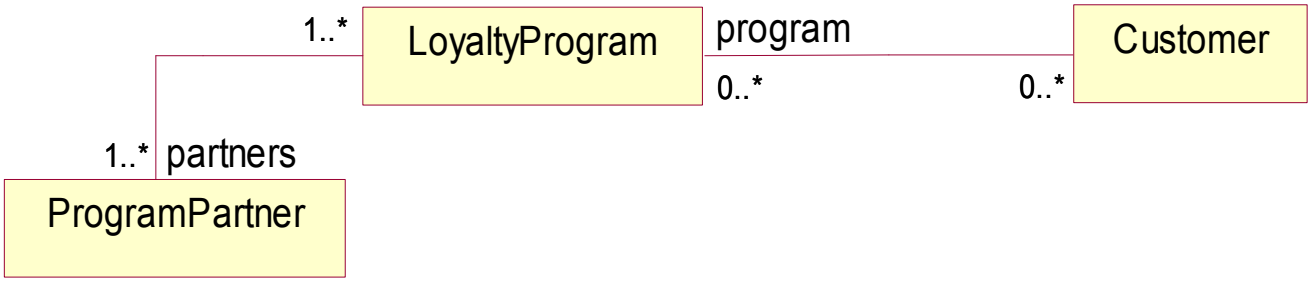
“Royal & Loyal” system

- כל חברה שמציעה ללקוחותיה חברות במועדון נקראת – *ProgramPartner*.
- כל תוכנית נאמנות יכולה להיות מוצעת על ידי יותר מחברה אחת.
- במקרה שתוכנית מוצעת על ידי מספר חברות, כל לקוח שחבר בתוכנית יוכל להנות מכל המבצעים שמציעות כל החברות בתוכנית.



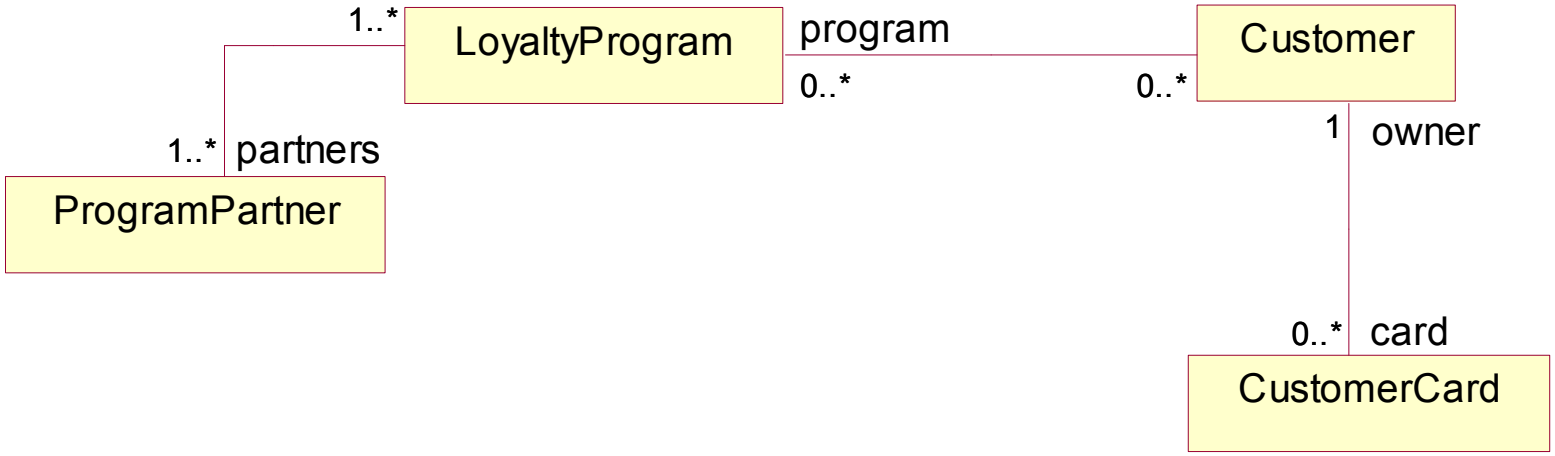
“Royal & Loyal” system

- כל לקוח – *Customer* - שמעוניין להצטרף כחבר במועדון, ממלא טופס חברות.



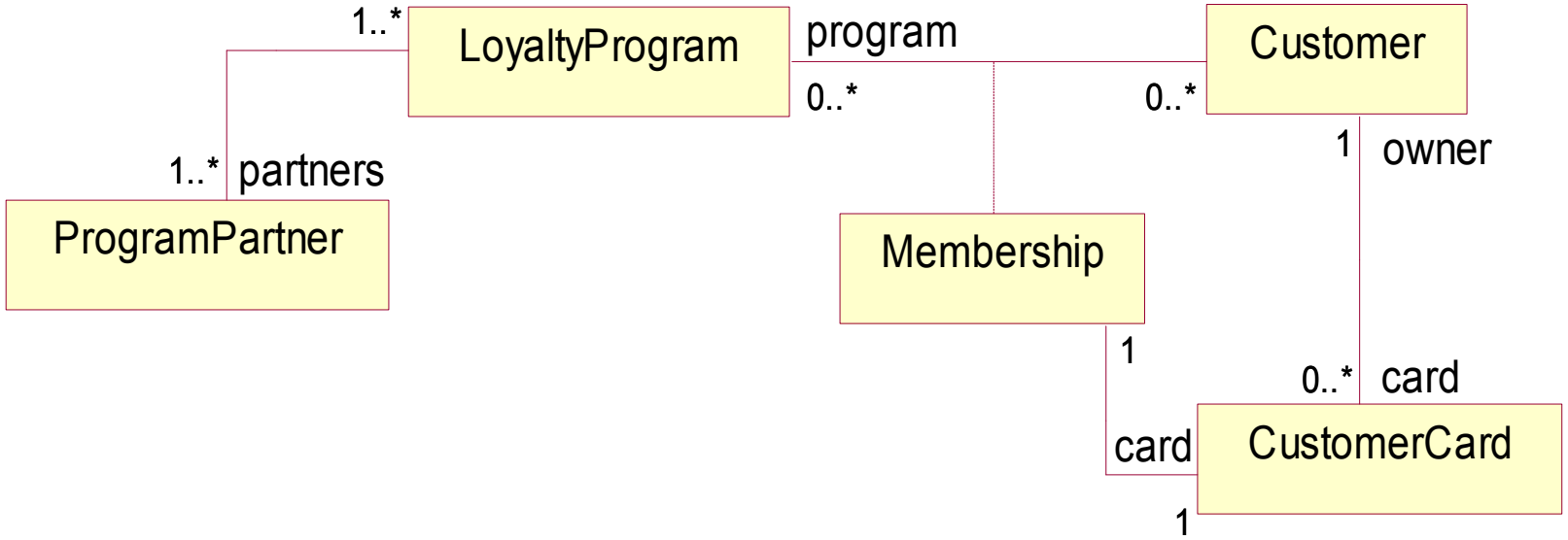
“Royal & Loyal” system

- כל לקוח שמצטרף כחבר לתוכנית מקבל כרטיס חבר - *CustomerCard*.
- כל כרטיס מונפק ללקוח יחיד.



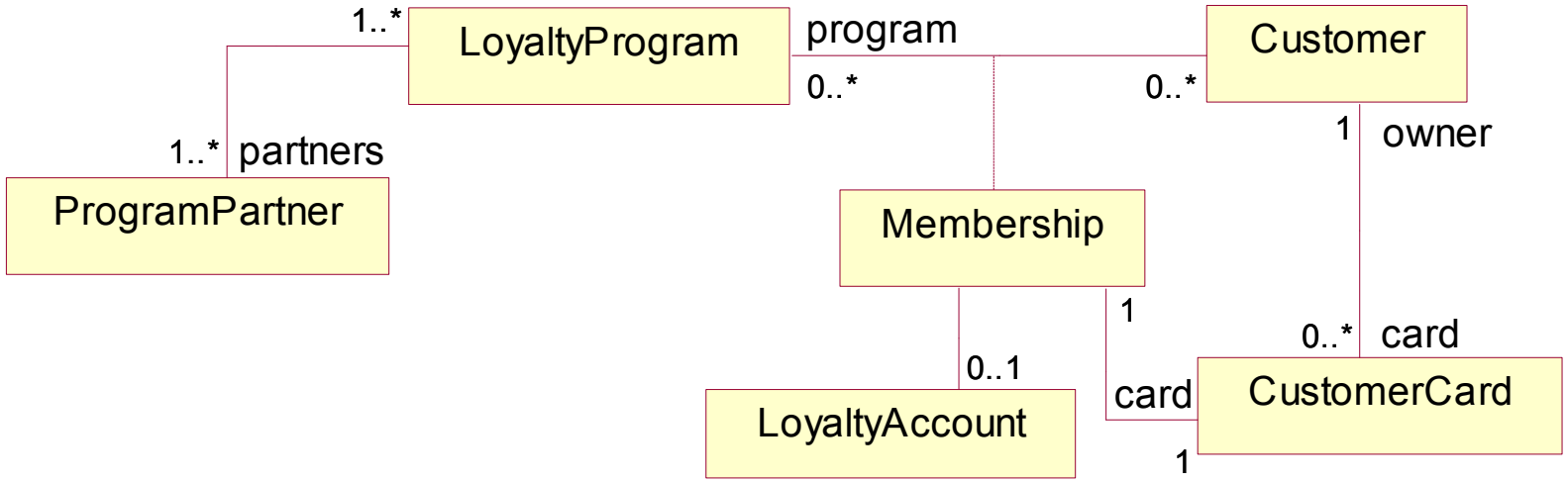
“Royal & Loyal” system

- לכל לקוח יכולים להיות מספר כרטיסים אבל לכל "חברות" – *Membership* -של לקוח בתוכנית יש כרטיס אחד בלבד.



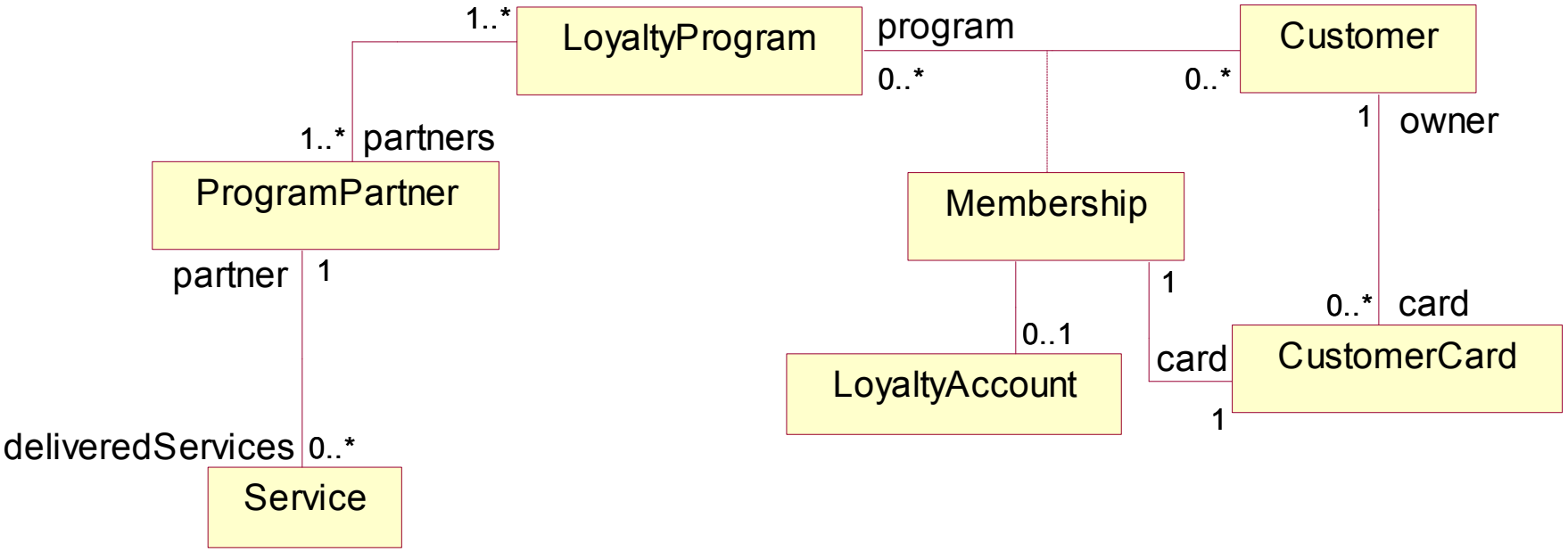
“Royal & Loyal” system

- מספר הנקודות נשמר לכל לקוח בעבור כל תוכנית בה הוא חבר.
- לפיכך, לכל לקוח יש חשבון לכל תוכנית – *LoyaltyAccount* – המכיל את מספר הנקודות שצבר הלקוח בתוכנית.



“Royal & Loyal” system

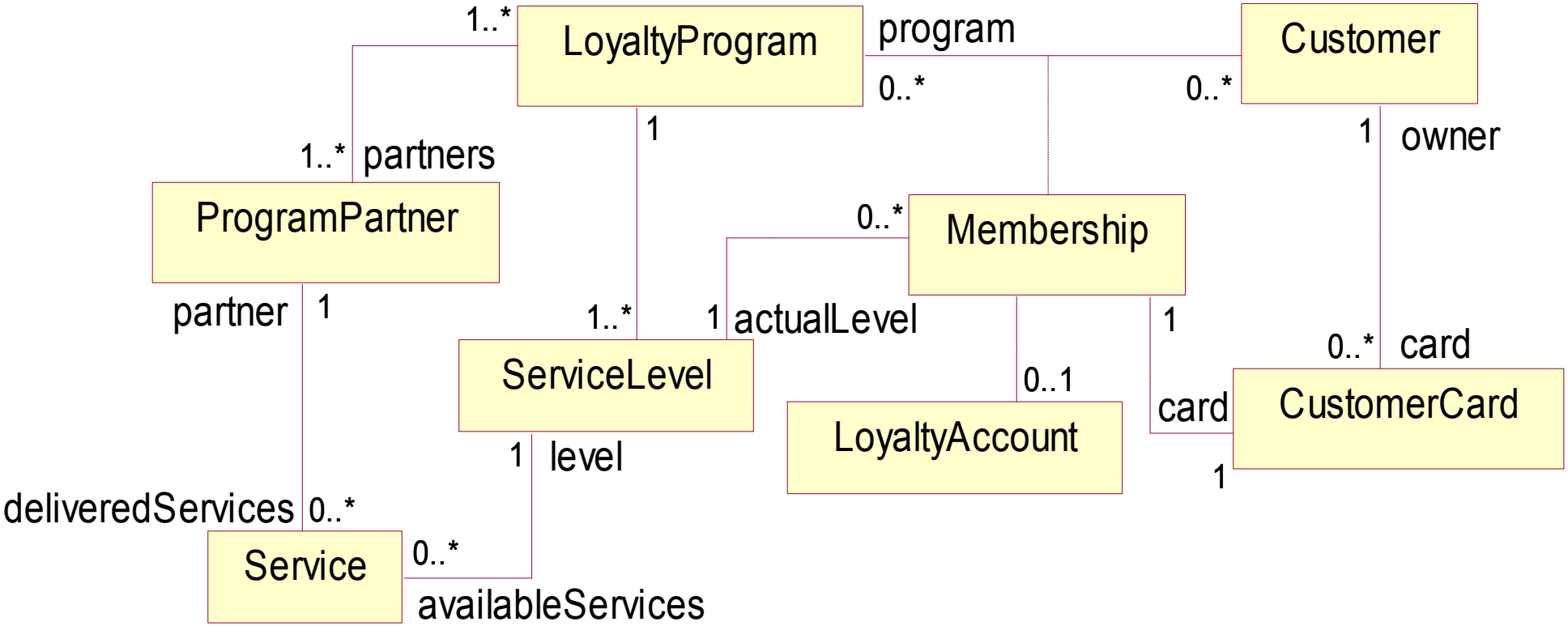
- כל חברה, שכלולה בתוכנית נאמנות, יכולה להציע מגוון שונה של שירותים ללקוח – *Service*.



“Royal & Loyal” system

- כל תוכנית שמוצעת ללקוח, יכולה להתחלק למספר רמות מנוי. ("מועדון הזהב","מועדון הכסף"...) – *ServiceLevel*
- כל רמה של מנוי יכולה לכלול אוסף אחר של שירותים המוצעים בתוכנית.
- כל רמת מנוי מוצעת לכל תוכנית בה חבר הלקוח – Membership.

“Royal & Loyal” system

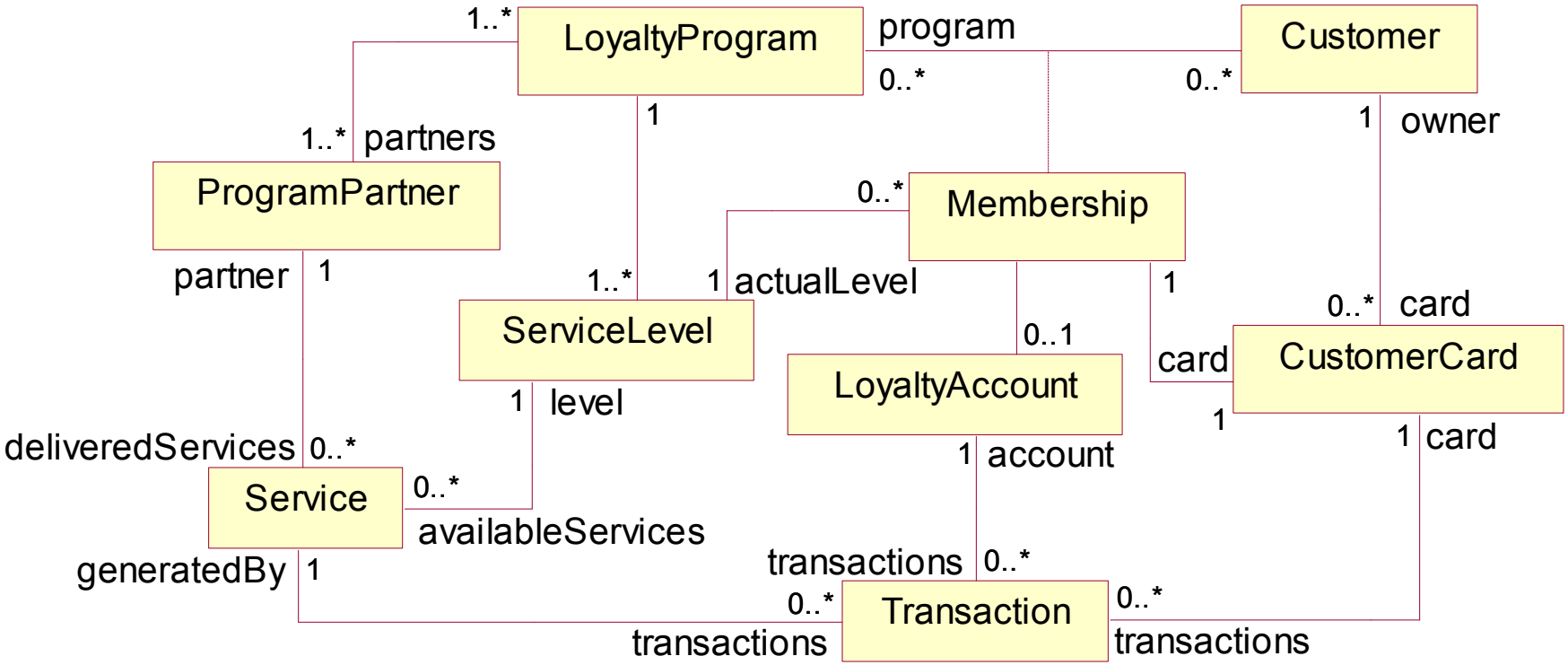


“Royal & Loyal” system

- לקוחות יכולים לבצע פעילויות שונות בכל מנוי:
  - לדוגמא, מועדון בשם "buy & bonus" מכובד על ידי : רשת היפר-נטו, דלק אלון, יורופקאר ואל-על.
  - השירותים הניתנים על ידי המועדון:
    - רשת היפר נטו – בונוס של 5 נק' לכל רכישה מעל 25ש"ח ואפשרות לקנות מוצרים באמצעות נק' שנצברו
    - רשת דלק אלון – הנחה של 5 ₪ בכל תדלוק
    - יורופקאר – בונוס של 20 נק' לכל 100 ₪ שמשולמים
    - אל על – 1 נק' בונוס מצברת לכל 15 מייל וטיסה חינם בתמורה לנק' כערך הכרטיס

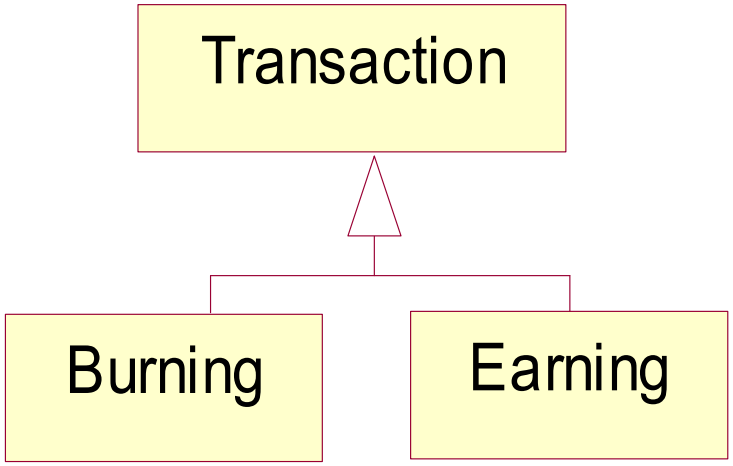
“Royal & Loyal” system

- החברה אחראית לתעד את כל הפעילויות שבוצעו בחשבון הלקוח – *Transaction*.

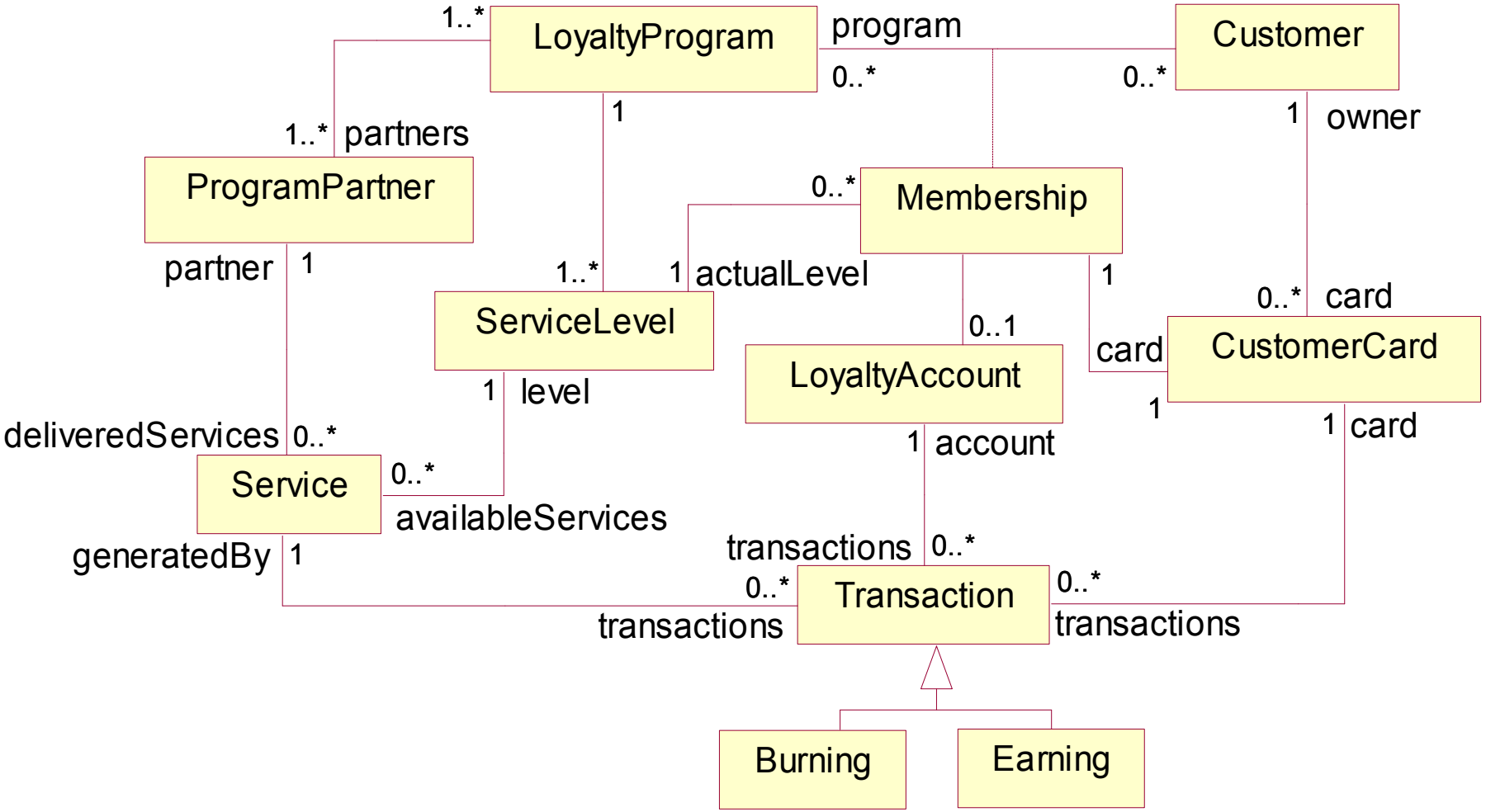


“Royal & Loyal” system

- ייתכנו 2 סוגי פעילויות ללקוח: צבירת נקודות – *Earning* – ומשיכת נקודות – *Burning*.



“Royal & Loyal” system – White Map



יצירת קוד ג'אבה מתרשים-מושגים

אפשר ליצור קוד-ג'אבה באופן אוטומטי מתוך תרשים-מחלקות של פאפירוס. לפרטים ראו כאן:
https://wiki.eclipse.org/Java_Code_Generation

אבל, זה תהליך מורכב ולא בטוח שהוא שווה את ההשקעה.

המטרה העיקרית של תרשים מחלקות היא לתאם בין המתכנתים לבין המנהלים ובין המתכנתים לבין עצמם – לא לכתוב קוד אוטומטית.