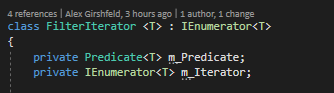
### 

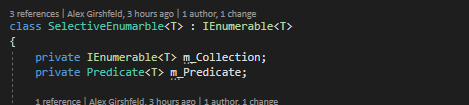
### תיאור קצר של הפיצ'רים שבחרנו לממש בתרגיל הקודם:

* ״Who likes me most?״ הינו פיצ׳ר המאפשר למשתמש לקבל רשימה המכילה את כל המשתמשים אשר אי פעם סימנו ״אהבתי״ (Like) לאחד או יותר מהפוסטים שפורסמו על ידו. בנוסף, הרשימה מסודרת לפי מספר הסימונים אותם עשה כל משתמש, ובכך מאפשרת למשתמש המבקש לראות מי הם המשתמשים אשר עוקבים אחר התוכן אותו הוא מפרסם ואוהבים אותו יותר מבין עוקביו.
* "Influence preservation grade per post" הינו פיצ׳ר המאפשר למשתמש לבחור פוסט ספציפי מבין כל הפוסטים אותם פרסם,ולקבל עליו ציון בין 0 ל-100 לאחר ניתוח השפעת אותו פוסט נבחר על קהל עוקביו של המשתמש. ככל שיותר משתמשים אשר עוקבים אחריו באופן קבוע סימנו Like לאותו פוסט, כך ציונו יהיה גבוה יותר בכך שעזר לשמר את השפעתו האינטרנטית של המשתמש על קהל עוקביו בכדי שימשיכו לעקוב אחר התוכן אותו הוא מפרסם.
* ״Influence expansion grade per post״ הינו פיצ׳ר משלים לנ״ל, המאפשר למשתמש לבחור פוסט ספציפי מבין כל הפוסטים אותם פרסם,ולקבל עליו ציון בין 0 ל-100 לאחר ניתוח השפעת אותו פוסט נבחר על קהל עוקביו של המשתמש. ככל שיותר משתמשים אשר אינם מסמנים Like באופן קבוע לפרסומים של המשתמש אכן סימנו Like לפוסט, ובפרט משתמשים אשר מעולם לא סימנו Like לאף פוסט של המשתמש, כך ציונו של הפוסט גבוה יותר, בכך שעזר להרחיב את השפעתו האינטרנטית של אותו משתמש ועזר לו להגיע לקהלים חדשים נוספים ובכך לאסוף משתמשים חדשים אשר יתווספו לקהל עוקביו.
* הוספת יכולת לעבוד ללא חיבור לאינטרנט - Offline Mode  
  על מנת לאפשר למשתמש יכולת עבודה גם כאשר איננו מחובר לאינטרנט, מימשנו בעזרת התבניות Adapter, Proxy, Facade כאשר ישנה אופציה של Remember Me אשר בעזרתה האפליקציה עושה שימוש ב-Cache ובך שומרת את נתוני המשתמש ונותן אפשרות להשתמש באפליקציה גם ב-Offline Mode.

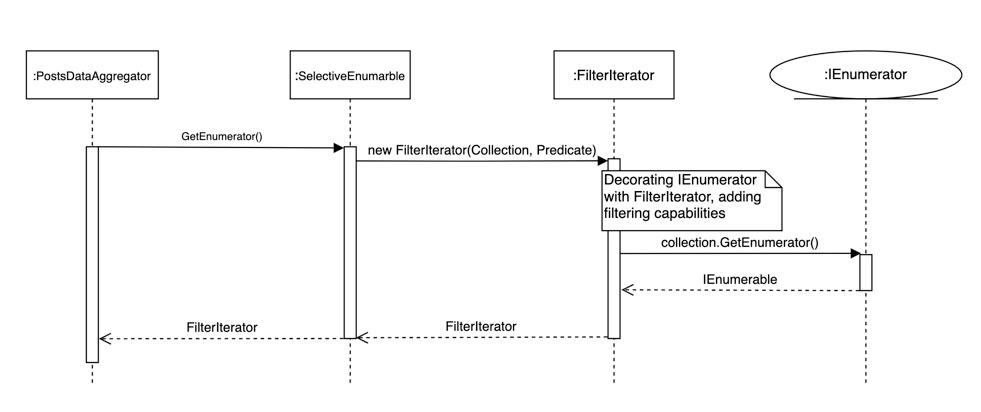
### תבנית מס' 1 – Decorator

* סיבת הבחירה / שימוש בתבנית:
  + תמיכה בתהליכי ניתוח מתקדמים כשרוצים לעבור רק על פריטים (פוסטים) מסויימים מתוך הAggregate:
    - במקרה שלנו השתמשנו לסנן פוסטים עם כמות לייקים גדולה מאפס.
  + ייצור תשתית Resuable לסקירה של עם הוספה של התנהגות "סינון"   
    ל-Aggregates.
  + אפשר להוסיף תנאים וסינון של אותם Aggregates בצורה "מרוכבת".
    - הוספה של סקירה "בררנית" עם מספר תנאים decorator של decorator.
* אופן המימוש:
  + אנחנו ממשים Decoratorים:
    - עבור IEnumarble וע"י SelectiveEnumarble.
    - עבור IEnumartor ע"י .FilterIterator





* Sequence Diagram:



* Class Diagram (דיאגרמה גדולה יותר בסוף המסמך)
  + אנחנו "מקשטים" (מוסיפים התנהגות סינון לEumartor) את הIEnumerable<T> בעזרת SelectiveEnumarble<T>
  + אנחנו "מקשטים" (מוסיפים התנהגות סינון) את IEnumerator<T> בעזרת FilterIterator<T>

A close up of a map

Description automatically generated

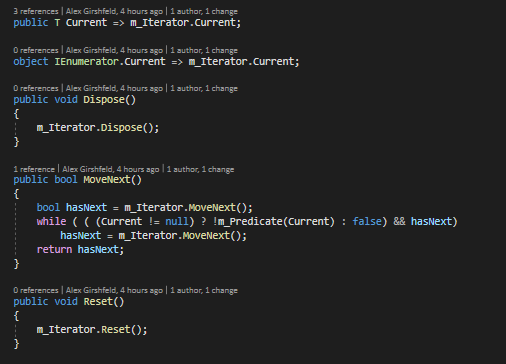
### תבנית מס' 2 – Iterator

* סיבת הבחירה / שימוש בתבנית:  
  אנחנו משתמשים בתבנית Iterator מכיוון שאנחנו רצינו לייצר תשתית Reusable לטובת סקירת Aggregates של אובייקטים ללא תלות באילו מבנה נתונים הוא ימומש בהמשך.

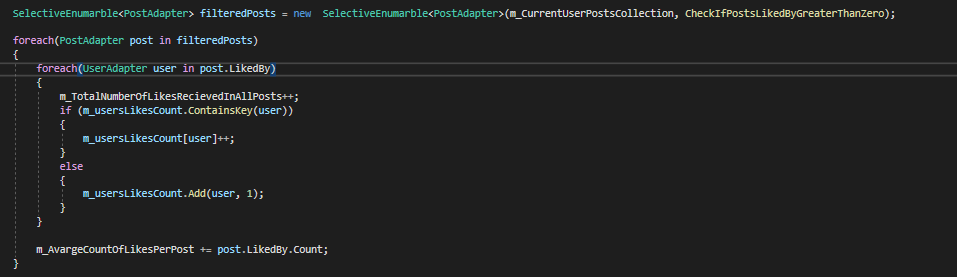
במהלך איסוף המידע מהאובייקטים רכיבים "סוקרים" Aggregates של אובייקטים, במקרה של הPostDataAggregator הוא עובר על Aggregate של AdapterPosts דרך לולאת foreach אשר משתמשת בIEnumartor לסקור את Aggregate.

* אופן המימוש:

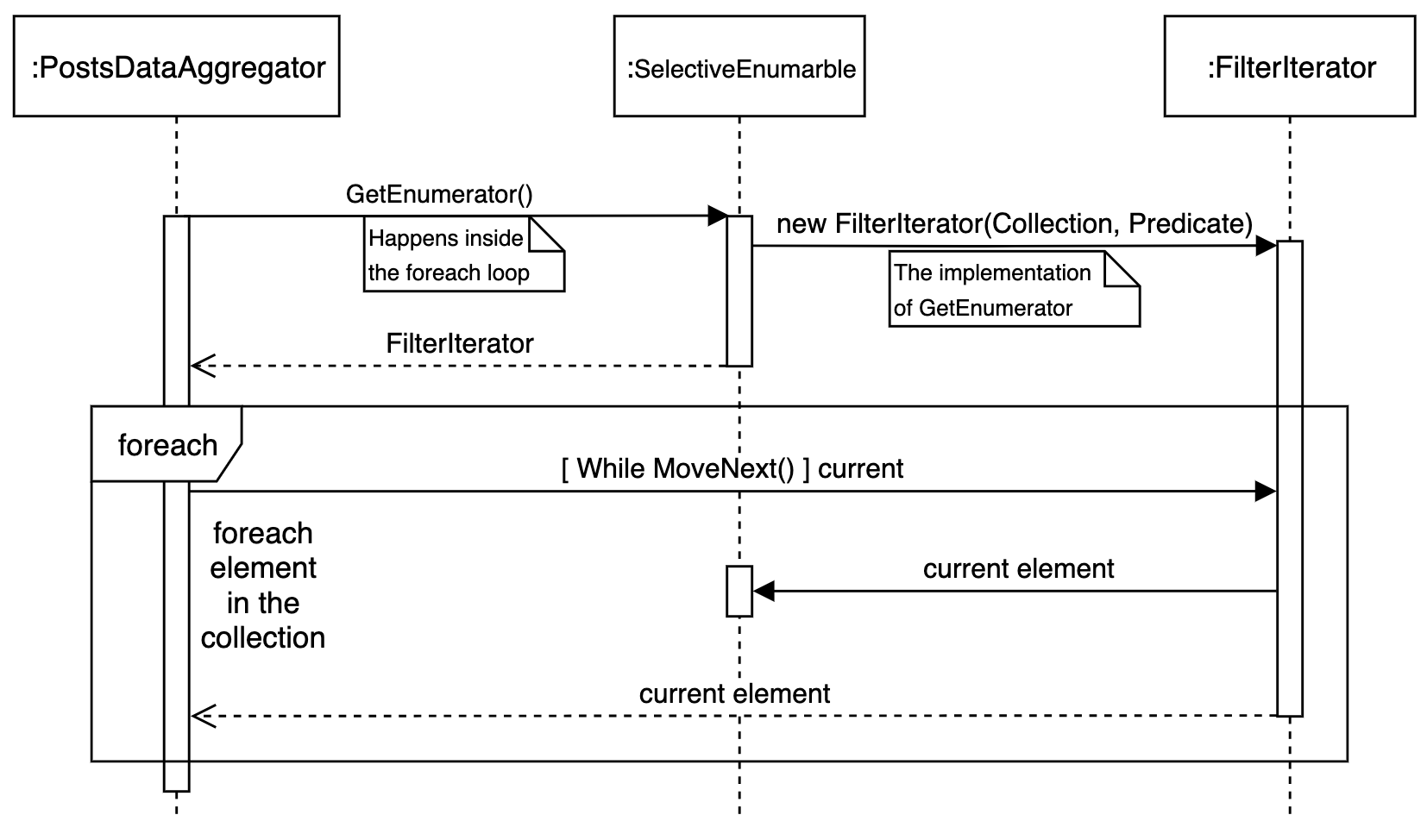
המחלקה FilterIterator ממשת את IEumerator:



והשימוש במחלקה זו קורה דרך GetEnumerator של SelectiveEnumarble (בדוגמה הזו זה בforeach):



* Sequence Diagram

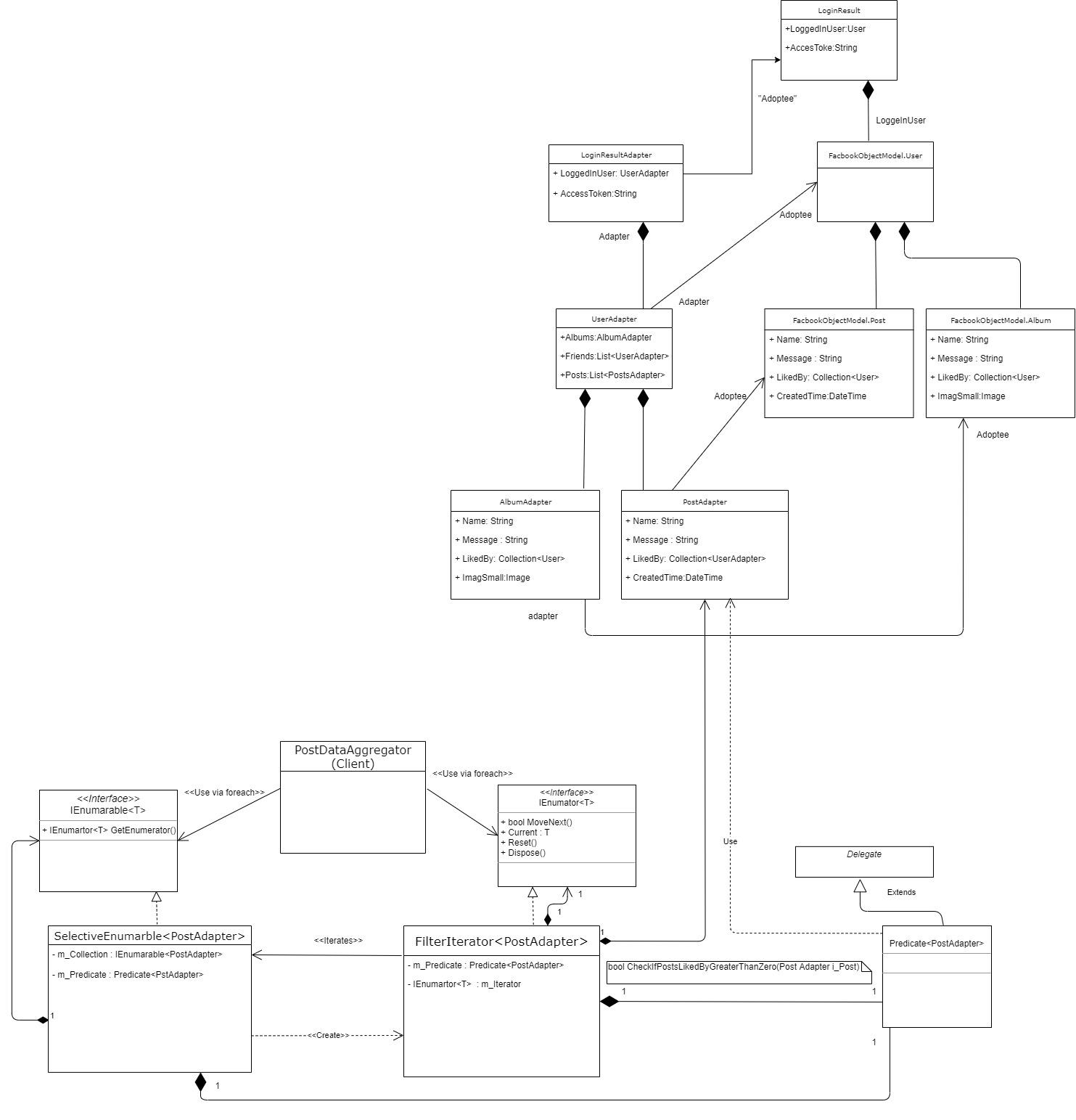


* Class Diagram (דיאגרמה גדולה יותר בסוף המסמך)

המחלקה FilterIterator היא הIterator שמימשנו

המחלקה SelectiveEnumarble היא המחלקה אשר מייצרת את הIterator (הAggregate)

המימוש של SelectiveEnumarble<T> הוא גנרי – המחשנו לטובת קריאות את השימוש עם PostAdapter



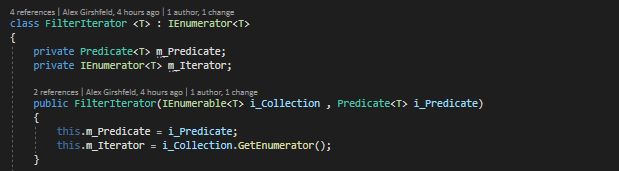
### תבנית מס' 3 – Strategy

* סיבת הבחירה / שימוש בתבנית:

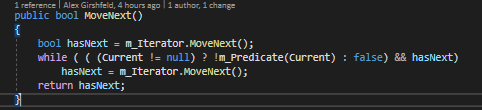
מימשנו Decoratorים שמוסיפים התנהגות "סינון" לAggregates של אובייקטים, כדי לאפשר שימוש חוזר עם לוגיקה משתנה לסינון השתמשנו בStrategy.

* אופן המימוש:

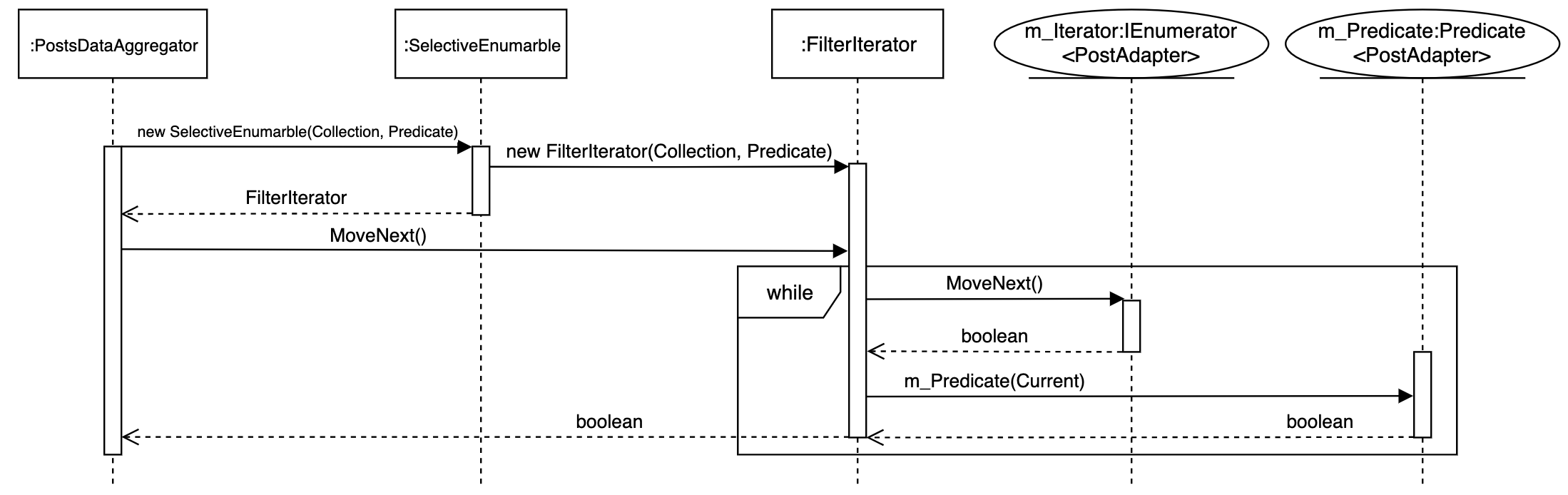
במהלך ייצור הDecoratorים עם התנהגות הסינון אנחנו מעבירים Predicate (Delegate שמחזיר בוליאני אם האובייקט עומד בתנאי מסוים)



השימוש בStrategy (Predicate):



* Sequence Diagram



* Class Diagram

במקרה שלנו הStrategy מכיל רק מתודה אחת שהיא הPredicate . הStrategy אשר מציין את תנאי הסינון עבור הAggregates הנ"ל הוא Delegate .

