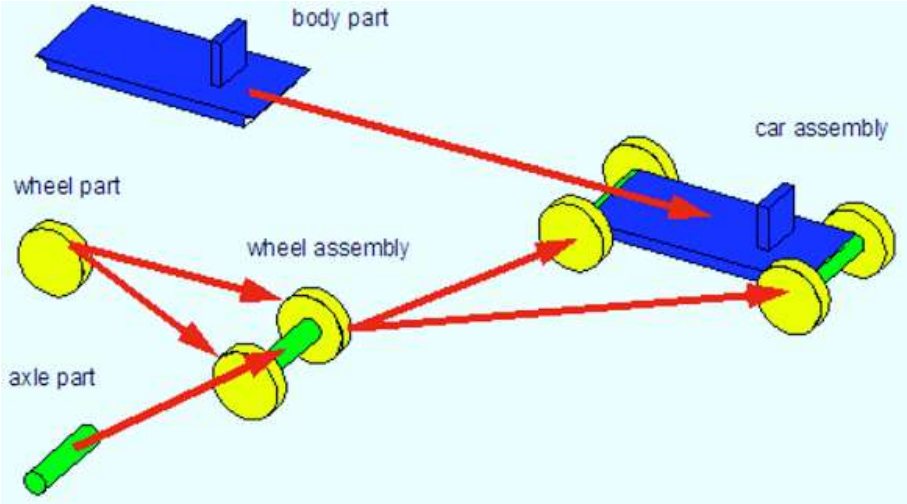
פרופ' עמיר תומר
amir@amirtomer.com

פרק 7

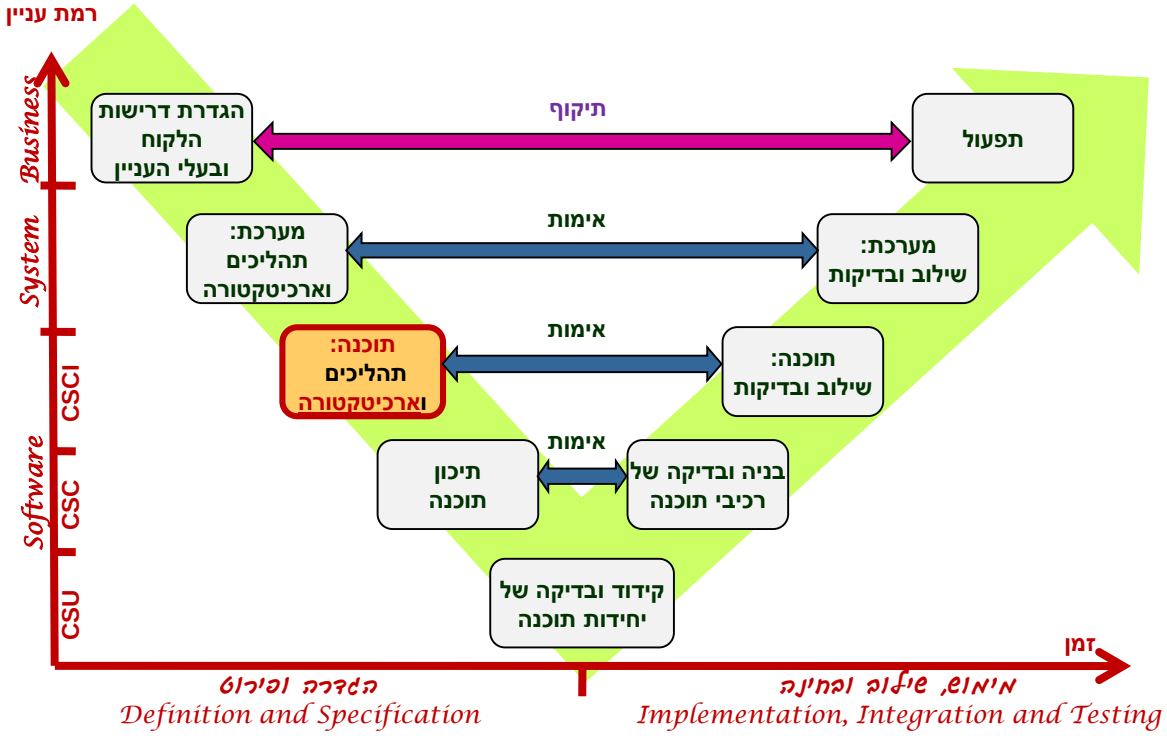
ארכיטקטורת התוכנה והמערכת
System and Software Architecture



Picture Source: http://www.techsoft3d.com/developers/technical-documentation/siemens-parasolid/

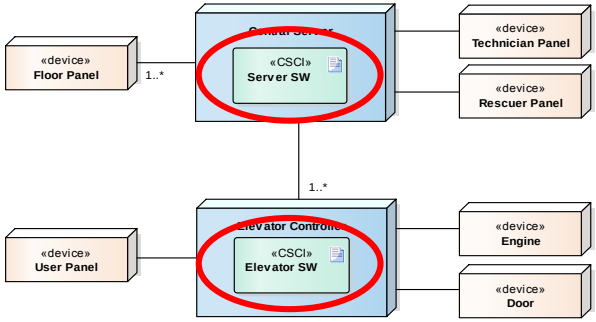
פעילות תיכון ארכיטקטורת התוכנה

- מטרת הפעילות
  - הגדרת המבנה (ארגון וממשקים) של רכיבי התוכנה
  - הגדרת התאימות בין ממשקי התוכנה (הלוגיים) לממשקי החומרה (הפיזיים)
- קלט
  - קבוצת רכיבי התוכנה ותהליכי התוכנה (sequence diagrams)
  - הארכיטקטורה הפיזית (Deployment Diagram)
- תוצרים
  - ארכיטקטורת תוכנה (Component Diagram)
  - ארכיטקטורת מערכת משולבת (Composite Diagram)

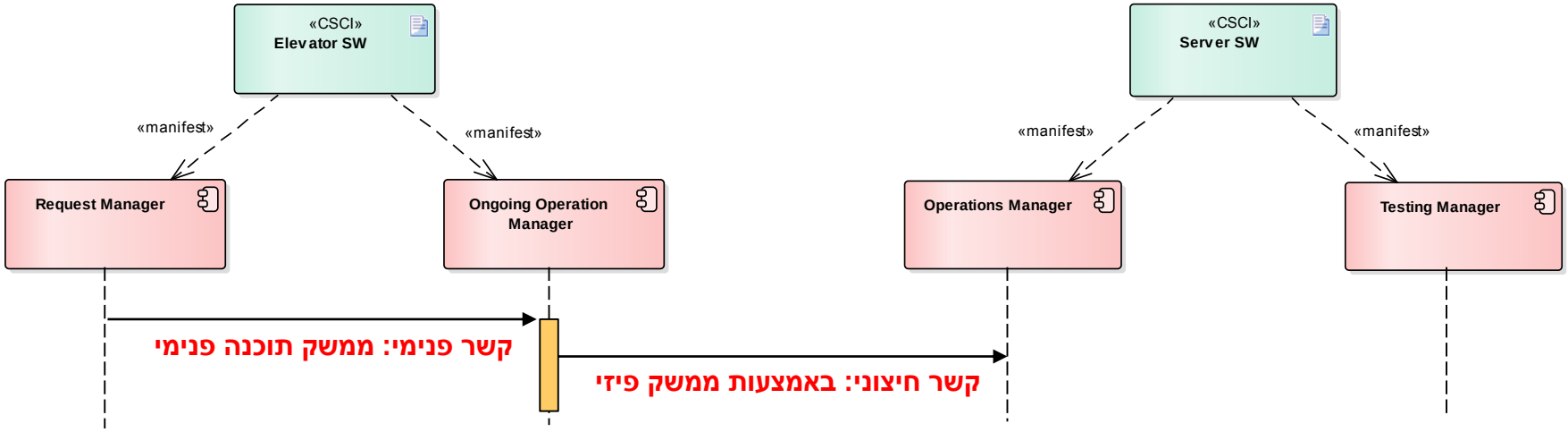


תמונת המצב עד כה

- בארכיטקטורה הפיזית קבענו את הקשרים הפיזיים בין צמתי החומרה וציינו את פריטי התוכנה המותקנים בהם



- באנליזה הפונקציונאלית פירקנו את פריטי התוכנה לרכיבים והקצינו להם פונקציות
- לאחר מכן קבענו את האינטראקציה בין הרכיבים הנדרשת למימוש ה-Use Cases

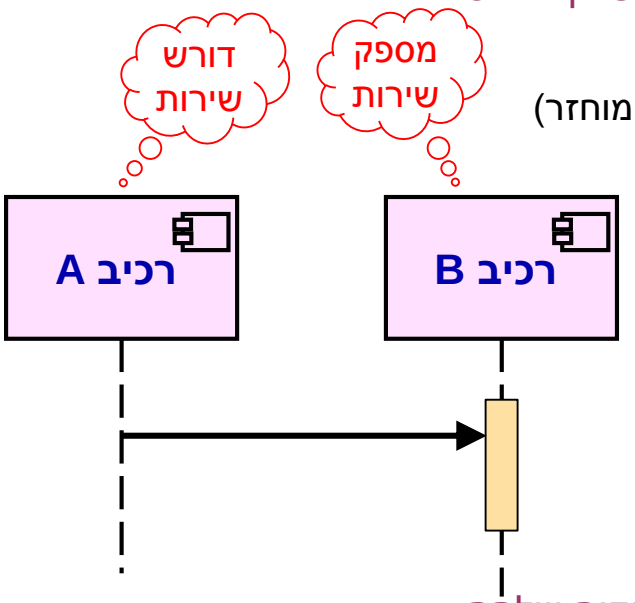


המעבר מארכיטקטורת מערכת (פיזית) לארכיטקטורת תוכנה (לוגית)

- ארכיטקטורת המערכת הממוחשבת הגדירה פירוק למרכיבים
  - פריטי חומרה: HWCI – HardWare Configuration Items
  - פריטי תוכנה: CSCI – Computer Software Configuration Items
- אין דואליות בין סוגי המרכיבים!
  - פריט חומרה הוא מרכיב עצמאי
    - יכול להתחבר ישירות לפריט חומרה באמצעות ממשק פיזי (ממשק חומרה-חומרה)
  - פריט תוכנה תמיד מותקן על גבי פריט חומרה
    - יכול להתחבר ישירות לפריט תוכנה (ממשק תוכנה-תוכנה) רק כאשר שני הפריטים נמצאים באותה חומרה
    - יכול להתחבר לפריטי תוכנה אחרים רק באמצעות הממשקים הפיזיים של החומרה (ממשק חומרה-תוכנה)
- במעבר מהארכיטקטורה הפיזית לארכיטקטורת התוכנה אנו "מקלפים" את החומרה, ומתייחסים לתוכנה כבנויה מרכיבים עצמאיים (Software Components)
  - בהמשך נחבר בין השתיים

ממשקים פונקציונאליים (ממשקי תוכנה)

- רכיב מקיים אינטראקציה עם סביבתו (רכיבים אחרים או ישויות חיצוניות) באמצעות שתי קבוצות של ממשקים:
- ממשקים מסופקים (Provided Interfaces)
  - קרי: ממשקים ל**שירותים המסופקים** ע"י הרכיב
    - האמצעים שחושף הרכיב לרשות סביבתו לצורך ביצוע הפונקציות שבאחריותו
    - דוגמאות
      - פונקציות ציבוריות (שם, פרמטרים וסוגיהם, סוג ערך מוחזר)
      - ממשק לשאילתות SQL
      - קליטת קוד הפעלה דרך עינית IR
      - קבלת קובץ
      - Dialog box
      - טופס HTML



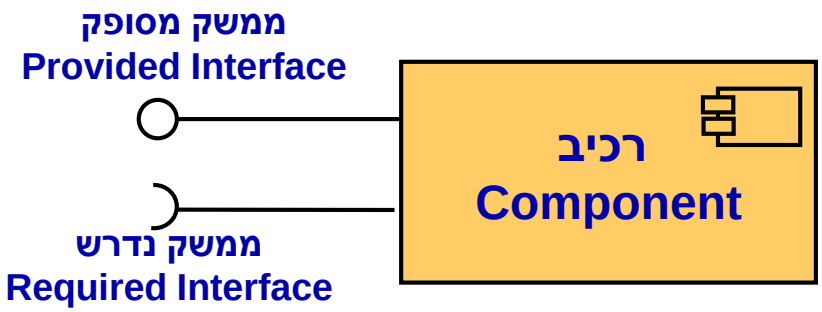
Application **P**rogram **I**nterface

- ממשקים נדרשים (Required Interfaces)
  - קרי: ממשקים ל**שירותים הנדרשים** ע"י הרכיב
    - פניות של הרכיב לרכיבים אחרים דרך הממשקים המסופקים שלהם
    - דוגמאות
      - אותן דוגמאות, מהצד השני (פרט לממשקי HMI)

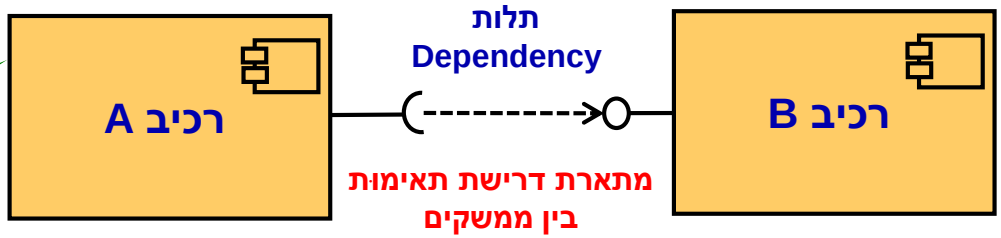
כיצד ממומשים הממשקים?

- כאשר הרכיבים הם מודולים באותה תוכנה (כלומר קומפילציה / קישור משותפים)
  - קריאה ישירה דרך API של מתודות
    - למשל שימוש בספריה סטטית
- כאשר הרכיבים נפגשים בזמן ריצה בתוך אותה סביבה (מחשב+מעה"פ)
  - קריאה דרך ממשק מוסכם
    - למשל שימוש בספריה דינמית DLL
- כאשר הרכיבים נמצאים בסביבות שונות
  - העברת הודעות דרך ממשק חומרה
    - למשל גלישה באתר ברשת באמצעות פרוטוקול HTTP באינטרנט

רכיבים וממשקים ב-UML



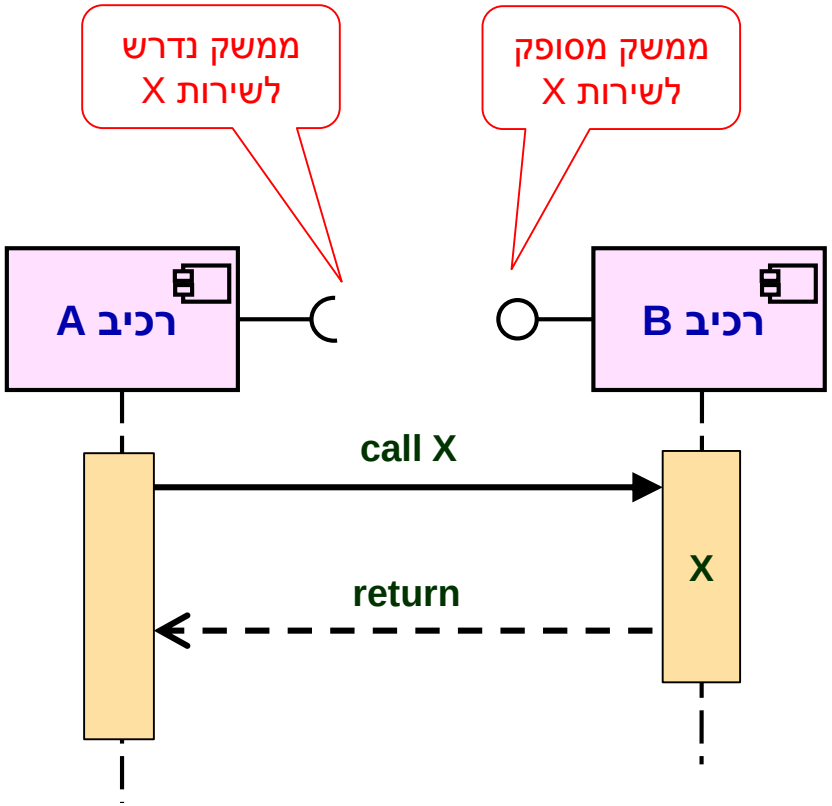
A תלוי ב-B פירושו:
שינוי ב-B עלול
לדרוש שינוי ב-A

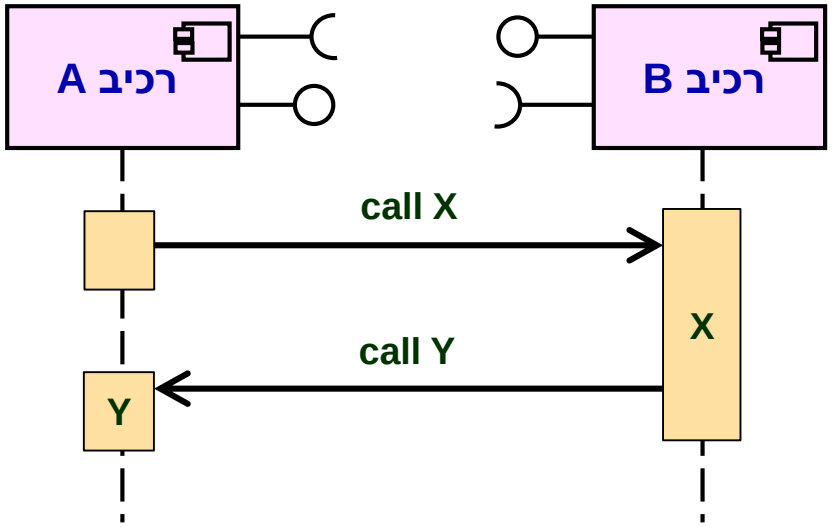


האם תמיד הממשק
הנדרש תלוי בממשק
המסופק?

זיהוי ממשקים של רכיב על בסיס האינטראקציה עם סביבתו

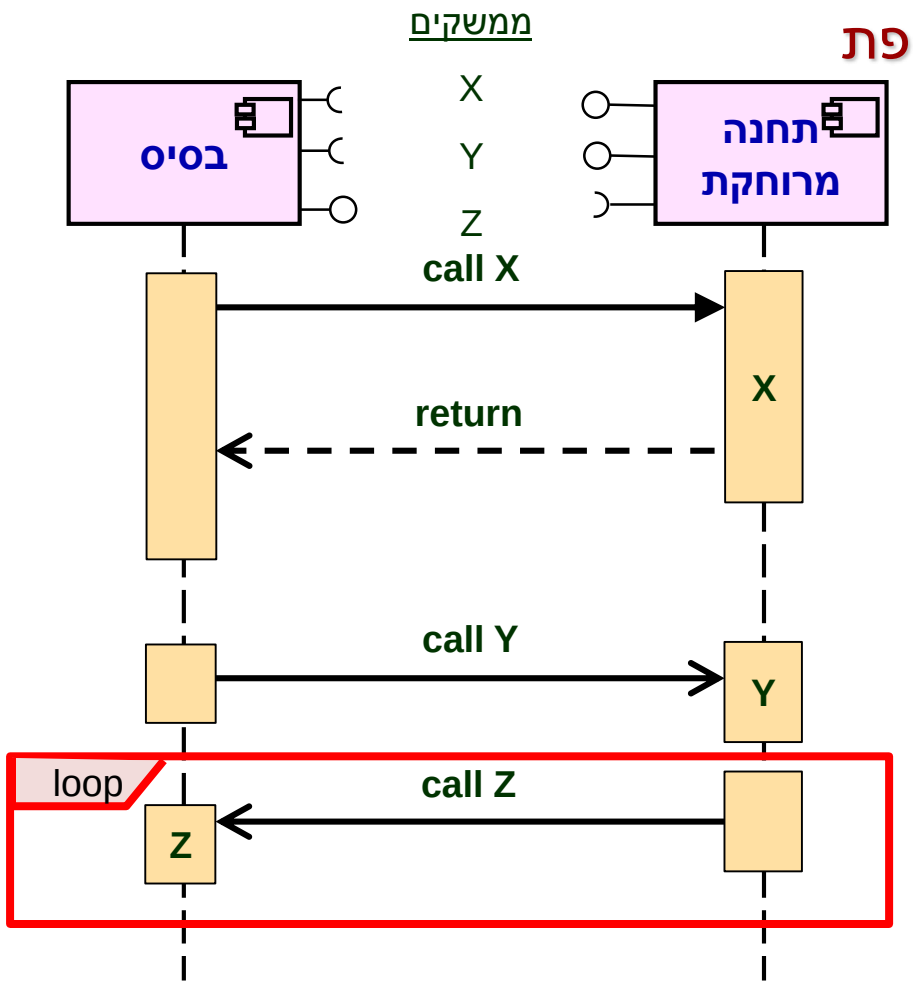
- בקריאה סינכרונית: התשובה מוחזרת כחלק מהקריאה (על גבי אותו ממשק)
- בקריאה א-סינכרונית: אין החזרת תשובה, ולכן אם B רוצה להודיע משהו ל-A הוא צריך ממנו שירות נוסף



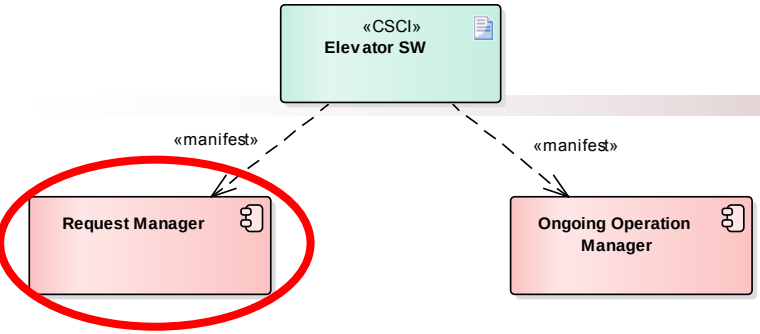


דוגמה קונקרטית

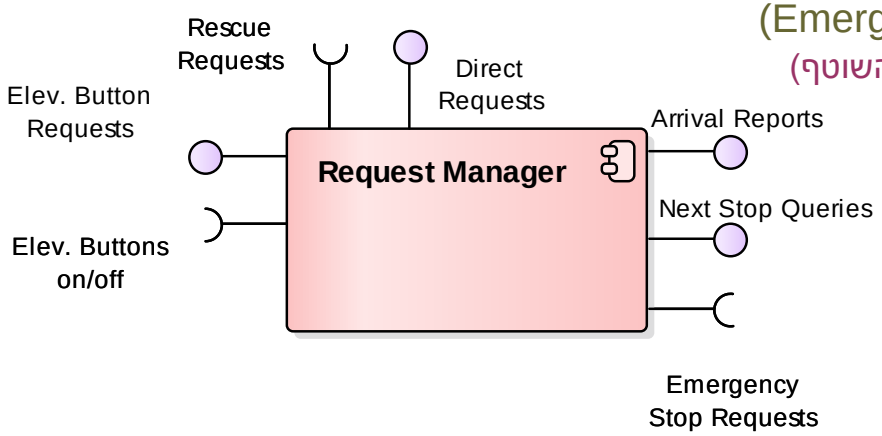
- X = בקשת נתון מיידי
- Y = הגדרת נתונים להעברה שוטפת
- Z = העברה שוטפת של נתונים



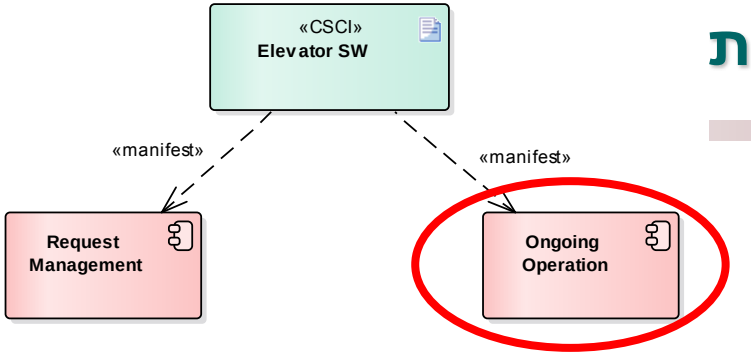
תוכנת מעלית – מנהל הבקשות



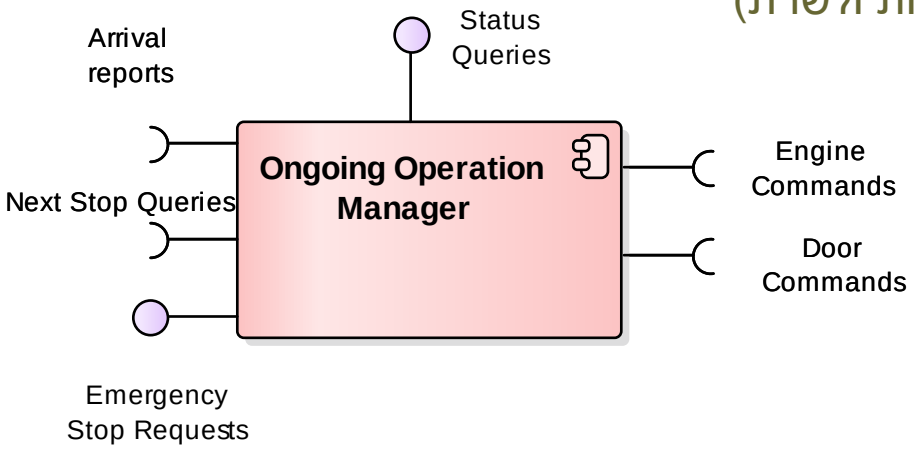
- **שירותים(פונקציות)**
  - זיהוי וטיפול/ניתוב בבקשות נסיעה, עצירה, חילוץ
  - ניהול רשימת המשימות של המעלית
- **ממשקים מסופקים**
  - בקשות מכפתורים בתוך המעלית (Elev. Button Requests)
    - ממשק לקליטת בקשות נסיעה, עצירת חירום והזמנת חילוץ המגיעות מכפתורי המעלית
  - בקשות ישירות (Direct Requests)
    - ממשק לקליטת בקשות נסיעה באופן ישיר (שלא באמצעות כפתורים) – מהטכנאי ומהפיקוד המרכזי
  - דיווח הגעה (Arrival Reports)
    - ממשק לקבלת הודעה שהמעלית הגיעה לקומה נתונה כדי למחוק את בקשת העצירה
  - שאילתות לגבי העצירה הבאה (Next Stop Queries)
    - ממשק לשליפת העצירה המיועדת הבאה, עבור רכיב התפעול השוטף
- **ממשקים נדרשים**
  - בקשת עצירת חירום (Emergency Stop Request)
    - פניה לבקשת עצירת חירום (מרכיב התפעול השוטף)
  - בקשת חילוץ (Rescue Request)
    - פניה לבקשת חילוץ (מהפיקוד המרכזי)
  - הדלקה/כיבוי כפתור
    - ממשק לפאנל הנוסע



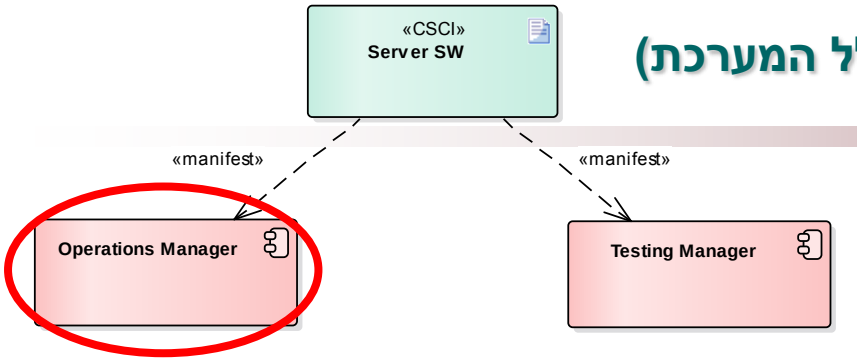
תוכנת מעלית – מנהל הפעילות השוטפת



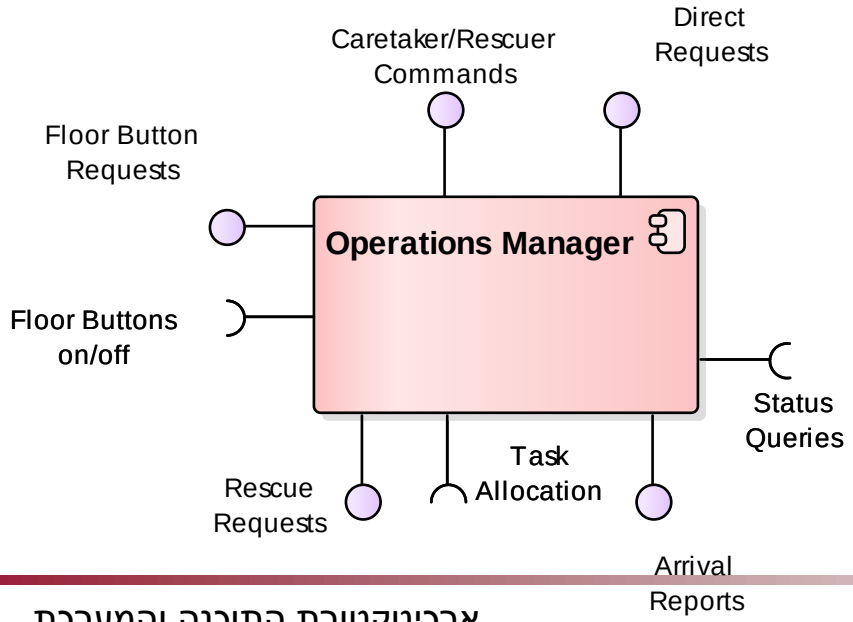
- שירותים(פונקציות)
  - ניהול הנסיעה השוטפת בין הקומות
  - עצירת חירום
- ממשקים מסופקים
  - בקשת עצירת חירום (מרכיב ניהול הבקשות)
  - בקשת סטטוס (מהשרת ומהטכנאי)
- ממשקים נדרשים
  - קבלת יעד העצירה הבא (ממנהל הבקשות)
  - דיווח על הגעה לקומה (למנהל הבקשות ולשרת)
  - פיקוד על המנוע
  - פיקוד על הדלת



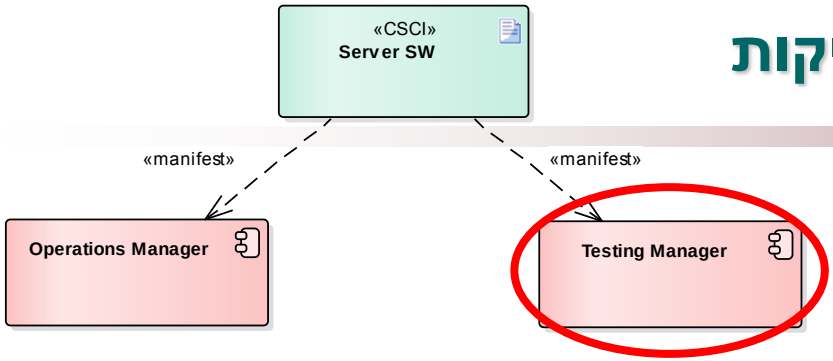
תוכנת השרת – מנהל הפעילות (של כלל המערכת)



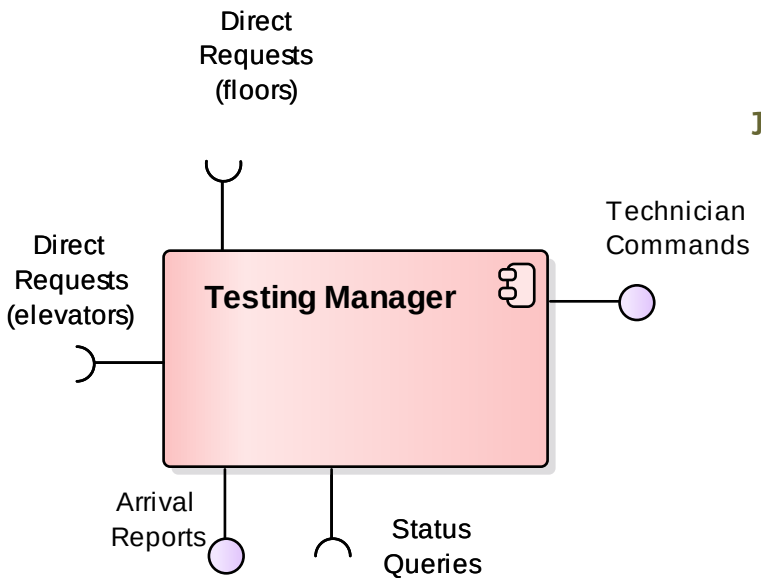
- שירותים (פונקציות)
  - זיהוי וניתוב בקשות מקומות
  - הקצאת מעליות
  - ניהול פעולות חילוץ
- ממשקים מסופקים
  - לחיצות על כפתורי קומות
  - הזמנות ישירות לקומות (מטכנאי) / מחלץ
  - פיקוד חילוץ ותחזוקה (פאנל מחלץ / איש אחזקה)
  - בקשות חילוץ (שהגיעו ממעליות)
  - קליטת דיווחי הגעה של מעליות לקומות (על מנת לכבות את כפתורי הקומות)
- ממשקים נדרשים
  - הדלקה/כיבוי של כפתורי קומות
  - בירור סטטוס של מעליות
  - הקצאת נסיעות למעליות



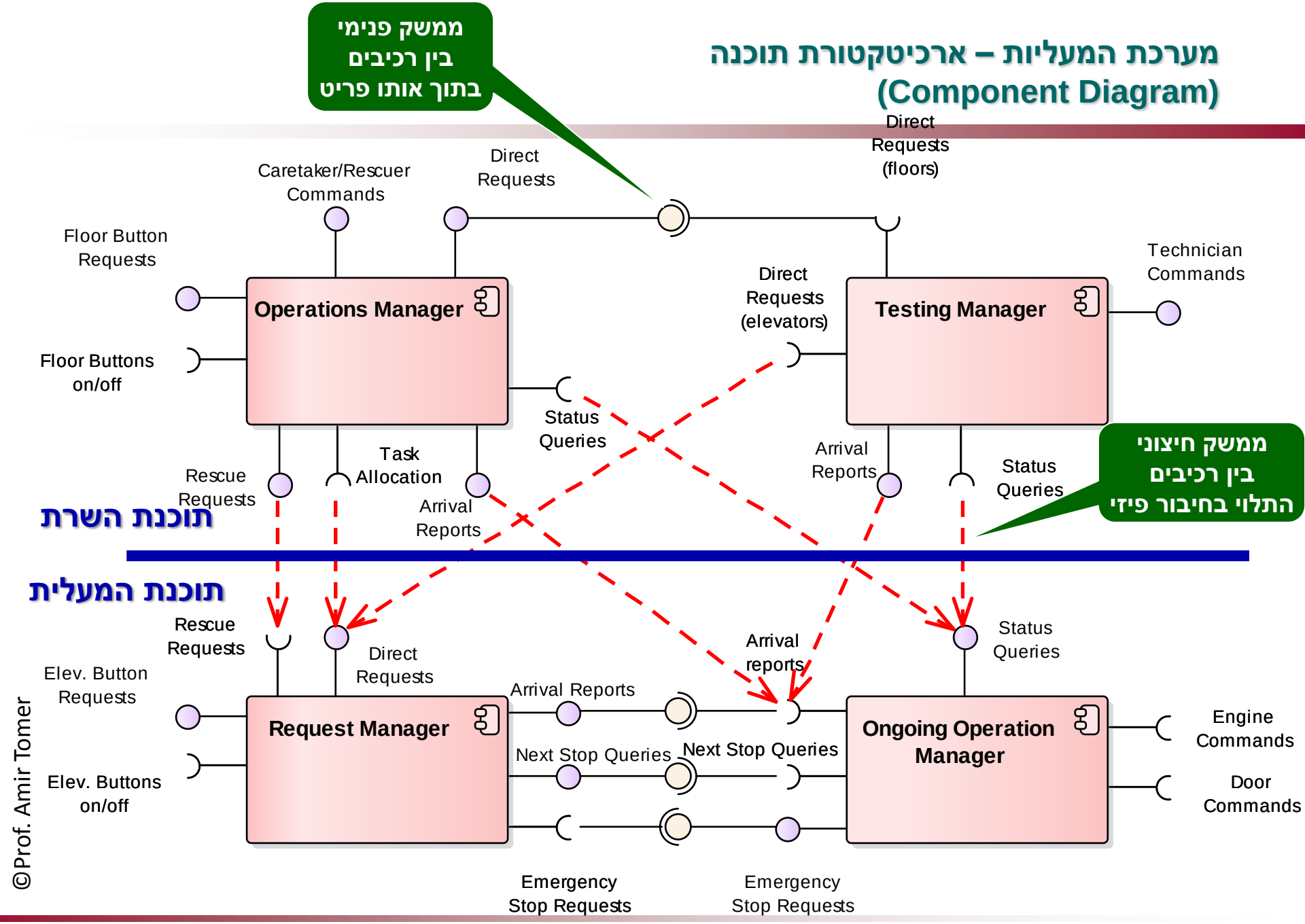
תוכנת השרת המעליות – מנהל הבדיקות



- שירותים (פונקציות)
  - זיהוי וניתוב פקודות טכנאי
  - ניהול בדיקה ותיקון של המערכת
- ממשקים מסופקים
  - פקודות טכנאי
  - קליטת דיווח על הגעת מעלית לקומה
- ממשקים נדרשים
  - בקשות נסיעה ישירות למעליות
  - בקשות הזמנה ישירות להגעת מעליות לקומות
  - בירור סטטוס של מעלית

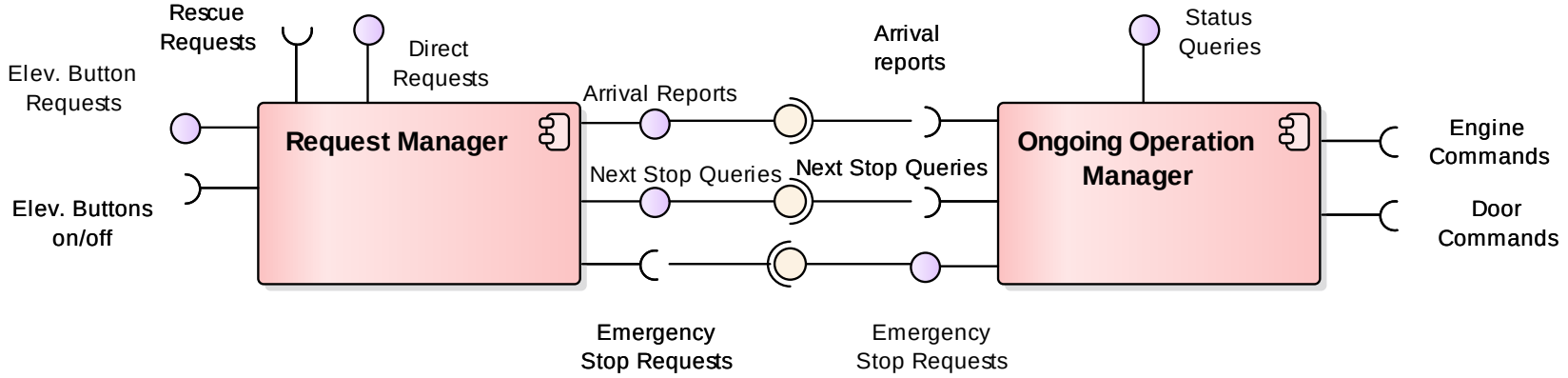


מערכת המעליות – ארכיטקטורת תוכנה (Component Diagram)



טבלת רכיבים וממשקי תוכנה (SW-ICD)

רכיבים			ממשקים			
זיהוי	שם	תפקיד	זיהוי	סוג	שירות	פירוט
RM	Request Manager	רישום וניהול בקשות העצירה מהמעלית	RMIF-1	מסופק	בקשות מכפתורים	העברת בקשות באמצעות לחיצה על כפתורים במעלית
			RMIF-2	מסופק	בקשות ישירות	העברת בקשות למעלית באופן ישיר
			RMIF-3	מסופק	דיווח הגעה	הודעה למנהל הבקשות שהמעלית הגיעה לקומה (כדי למחוק את בקשת העצירה)
			RMIF-4	מסופק	שאילתת העצירה הבאה	מהי התחנה הבאה בה אמורה המעלית לעצור
			RMIF-5	נדרש	הדלקת/כיבוי כפתור	הדלקה או כיבוי של כפתור הקומה במעלית
			RMIF-6	נדרש	סטטוס מעלית	בקשת סטטוס מעלית
			RMIF-7	נדרש	בקשת חילוץ	העברת בקשת חילוץ לשרת המרכזי
			RMIF-8	נדרש	בקשת עצירת חירום	העברת בקשת עצירת חירום למנהל הפעילות השוטפת של המעלית
OO	Ongoing Operation Manager	ניהול הפעולה השוטפת של המעלית	OOIF-1			
			OOIF-2			
			OOIF-3			
			OOIF-4			
			OOIF-5			



מטלה: ארכיטקטורת תוכנה

- יש לבנות ארכיטקטורת תוכנה עבור כל אחד מפריטי התוכנה של ePark
  - פיתחו Component Diagram חדש
  - גררו לתרשים את רכיבי התוכנה המרכיבים את אחד הפריטים (על בסיס קשרי <<manifest>> שהגדרתם)
  - הגדירו ממשקים מסופקים ונדרשים לכל רכיב
  - קשרו את הממשקים המתאימים באמצעות Assembly Connector
  - חיזרו על הפעולה עבור פריטי התוכנה האחרים
  - חברו את הממשקים מתאימים בין הפריטים באמצעות קשרי תלות

הממשקים הפונקציונאליים עוברים דרך הממשקים הפיזיים

ממשקים פיזיים של סמארטפון

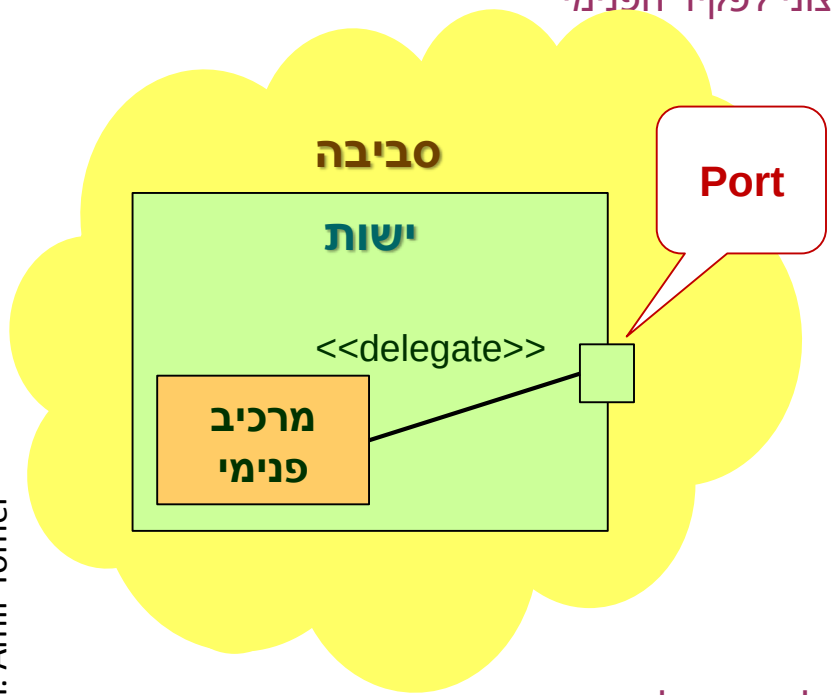
ממשקים פונקציונאליים של אפליקציית ניווט בסמארטפון

Provided
Interfaces

Required
Interfaces

האצלה (delegation) של ממשקים בישויות מורכבות

- במבנה המערכתי החיבור בין הסביבה או ישות חיצונית לבין ישות או מרכיב פנימי נעשה באמצעות ישות ביניים
  - לדוגמה, לקוח נותן הוראה לפקיד הבנק באמצעות מערכת הטלפון
    - הטלפון משמש כאמצעי חיבור בין הלקוח החיצוני לפקיד הפנימי

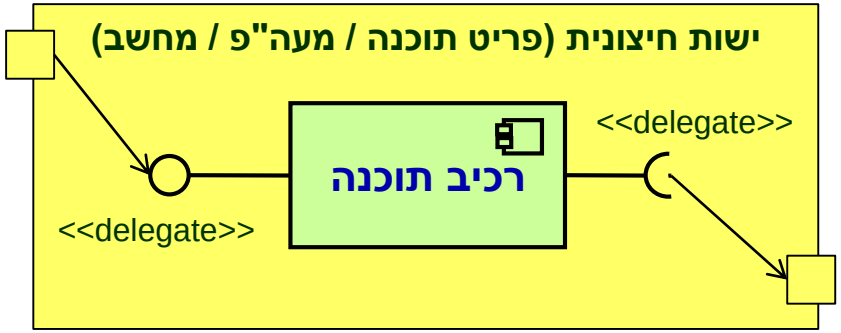


- יציאה (Port)
  - נקודת חיבור בין ישות לסביבתה
    - דוגמה 1: תקע/שקע או כל מחבר אחר
    - דוגמה 2: צומת ברשת תקשורת
    - דוגמה 3: פונקציות I/O של מערכת ההפעלה

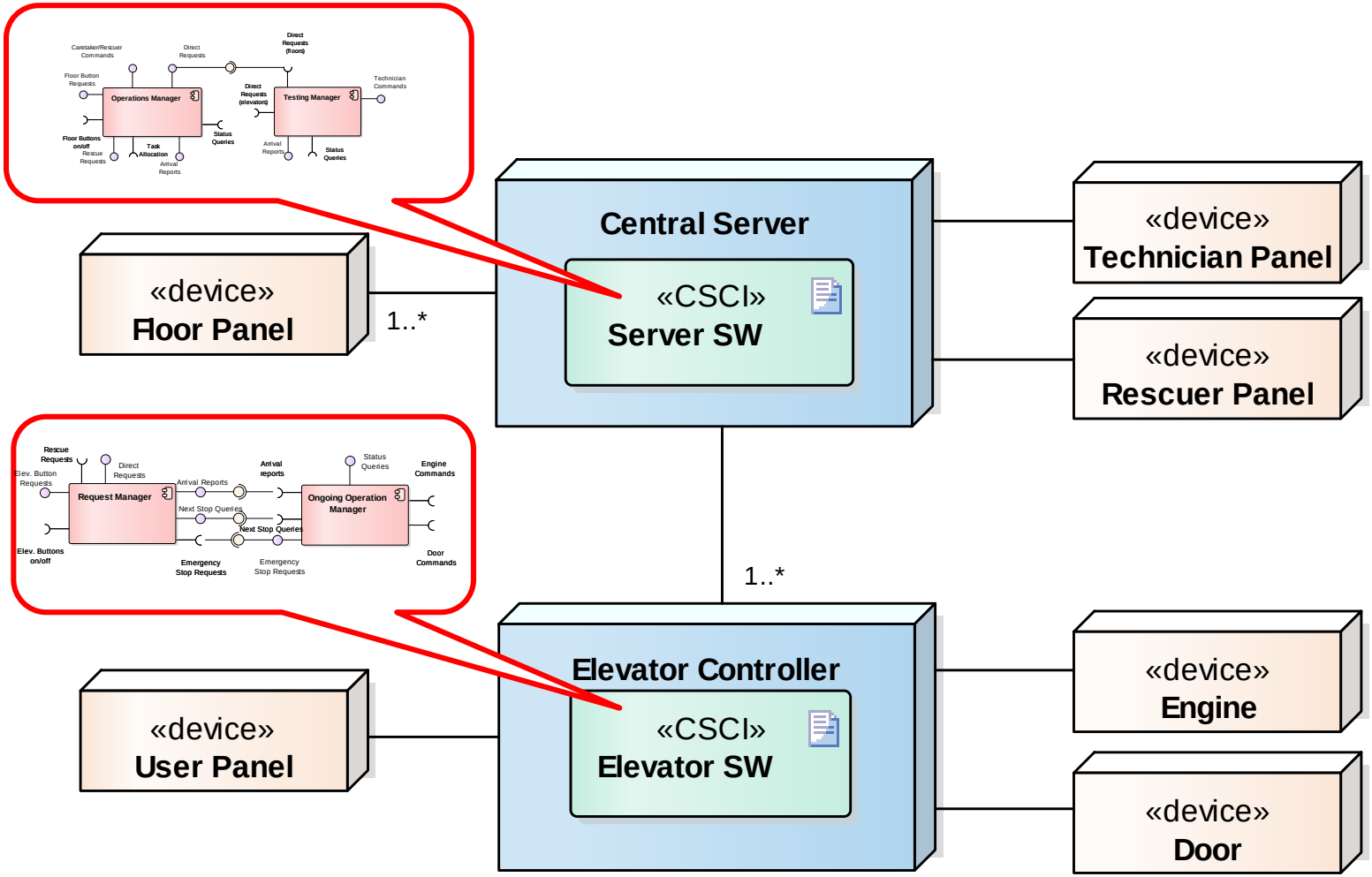
- חיבור האצלה (delegation connector)
  - חיבור בין יציאה חיצונית לבין מרכיב פנימי
    - חיבור יוצא: המרכיב הפנימי **משתמש** ביציאה לצורך קבלת שרות מהסביבה
    - חיבור נכנס: המרכיב הפנימי **מממש** את השרות הניתן דרך היציאה

מימוש ממשקי התוכנה באמצעות חיבורי החומרה

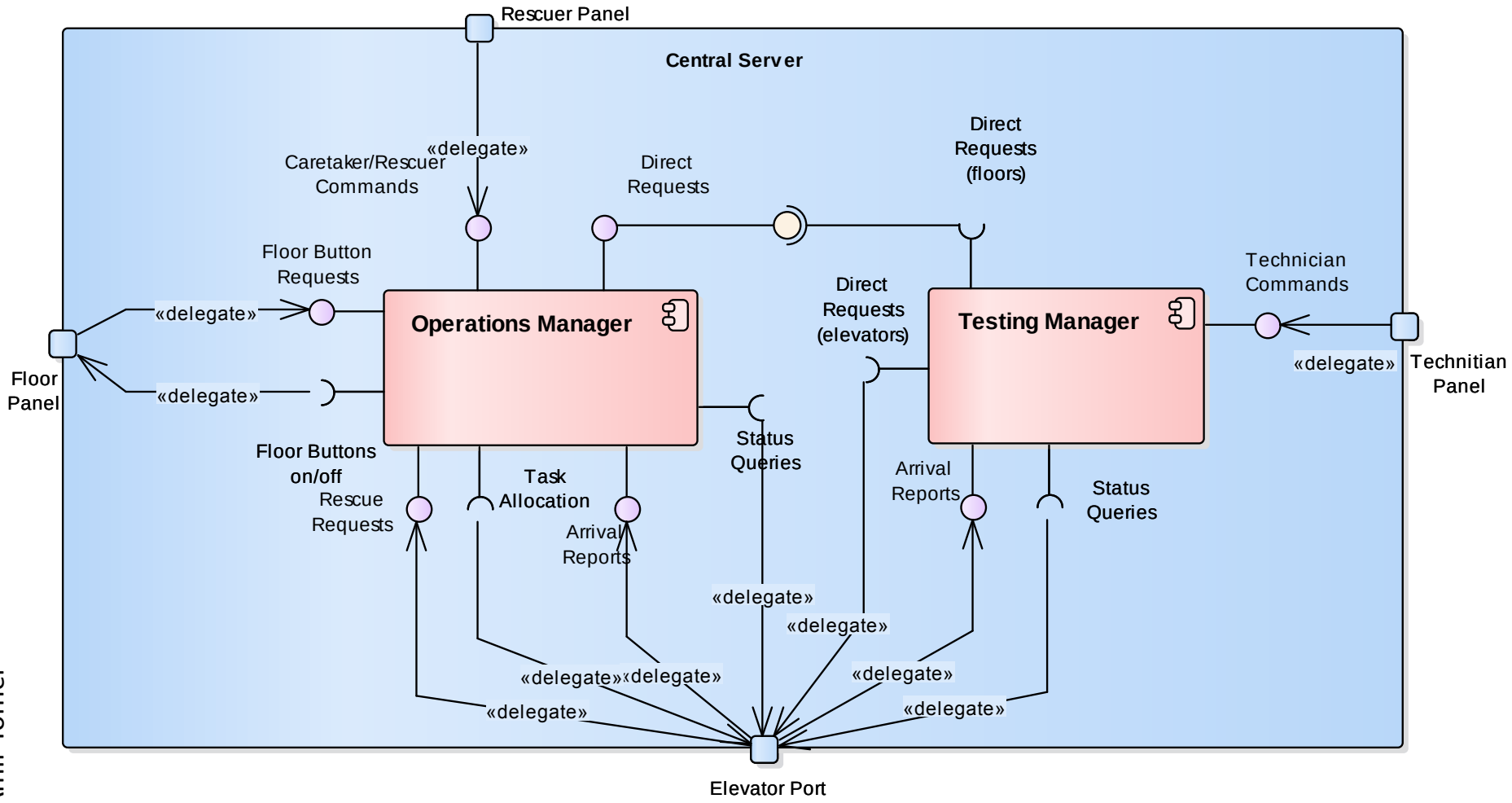
- כאמור, תוכנה תמיד "עטופה" בחומרה
  - פריט תוכנה תמיד מותקן בפריט חומרה (צומת)
  - ממשק בין פריט/רכיב תוכנה לסביבה החיצונית עובר תמיד דרך יציאה/מחבר של חומרה
- רכיבי תוכנה משתמשים ביציאות (ports) לצורך קשר עם סביבתם
  - חיבורי התקנים חיצוניים למחשב (USB, יציאות תקשורת וכו')
  - חיבורים דרך מערכת ההפעלה / התקשורת (קבצים חיצוניים, קלט/פלט, פרוטוקולי תקשורת)
  - חיבורים דרך אמצעים אחרים של הפריט (item) המייצג אותם (דרייברים וכו').
- האצלת ממשק מסופק
  - הרכיב מממש את השרות הניתן לסביבה החיצונית דרך היציאה
- האצלת ממשק נדרש
  - הרכיב מקבל את השרות מהסביבה החיצונית דרך היציאה



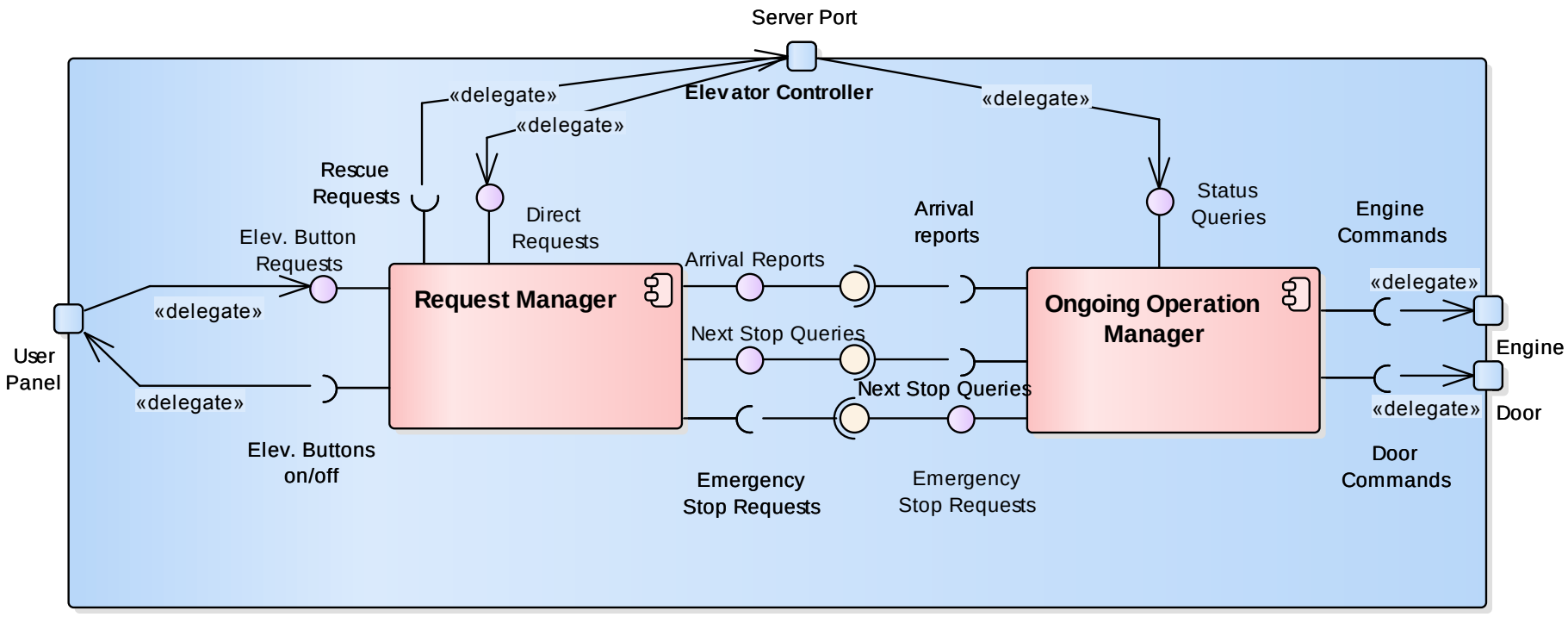
מערכת המעליות –
שילוב הארכיטקטורה הלוגית עם הארכיטקטורה הפיזית



מערכת המעליות – ארכיטקטורת מערכת כוללת – שרת מרכזי

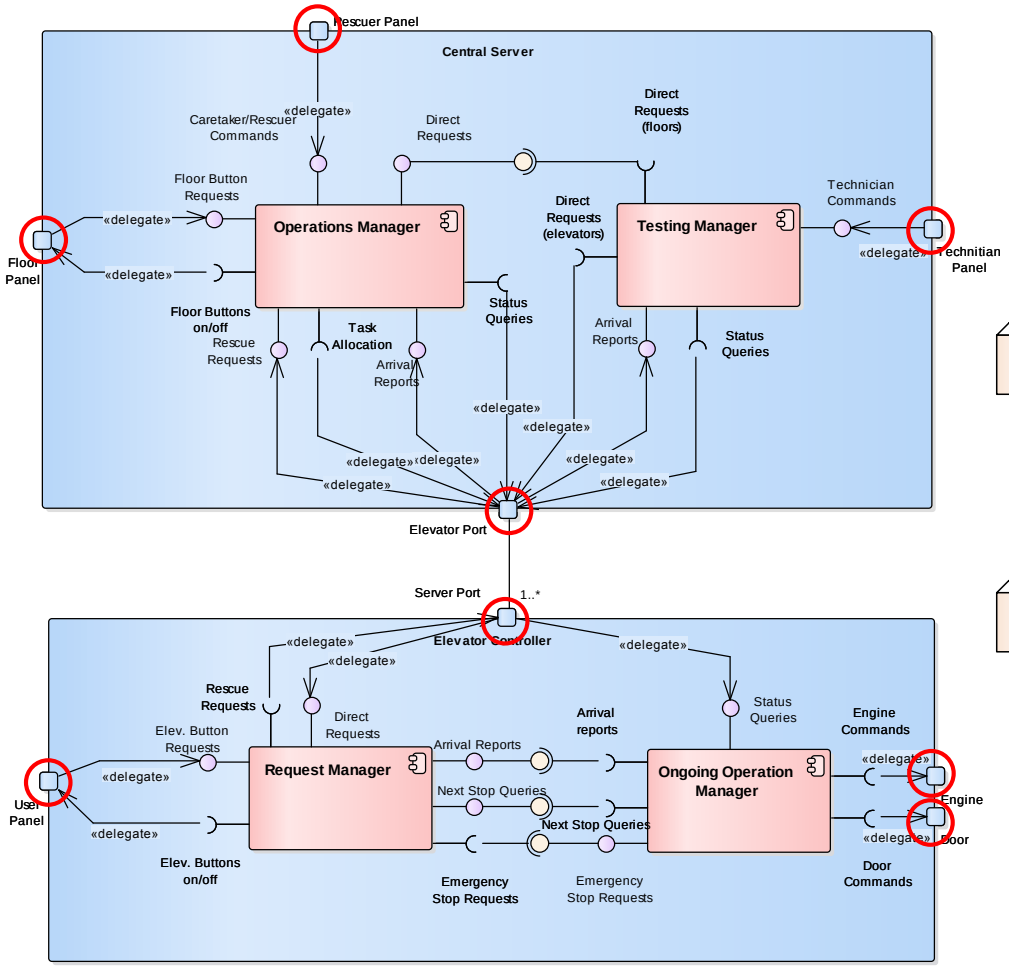


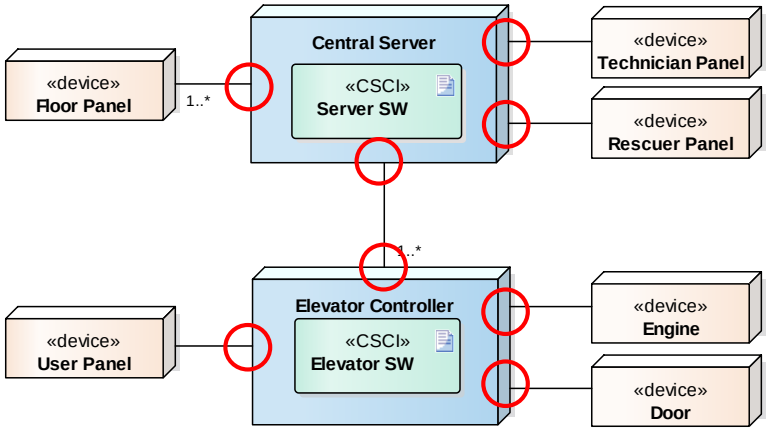
מערכת המעליות – ארכיטקטורת מערכת כוללת – בקר מעלית



שמירת עקביות (consistency) בין ארכיטקטורת התוכנה לארכיטקטורת החומרה

1. לכל ממשק חיצוני של ארכיטקטורת התוכנה יש port המייצג אותו (delegate)
2. כל port בתרשים ארכיטקטורת התוכנה מזוהה עם קשר פיזי בתרשים הפריסה (deployment diagram) של ארכיטקטורת המערכת





מטלה: ארכיטקטורת מערכת כוללת

- יש להגדיר ארכיטקטורת מערכת כוללת עבור ePark
  - פיתחו תרשים חדש מסוג Composite Diagram
  - גררו כל אחד מפריטי התוכנה (Software Artifacts) שבארכיטקטורה הפיזית אל התרשים, אך הגדירו כל אחד כ-part
  - לכל part הגדירו ports המתאימים לממשקים הפיזיים שלו
  - העתיקו והדביקו את ארכיטקטורת התוכנה של כל פריט אל תוך ה-part שיצרתם
  - קשרו את הממשקים החשופים ל-ports המתאימים באמצעות קשרי delegate