			Eventing. יכולה להיות בעיה במקרה של עבודה ללא ישות מרכזית בסביבה טקטית.	
5	פיתוח פשוט על בסיס חשיפת ממשקי תשתית ה-Grid: Grid-RPC, תמיכה בטרנזקציות, שירותי Workflow.	2	יש צורך בפיתוח ייעודי על מנת לממש את התהליך בצורה מלאה.	שירותי אירוח השירותים
4	מתבסס על פתרונות *WS/	4	מענה חשיפת המידע והצגתו על-ידי גורם שלישי באמצעות WSRP (Remote Portals). מימוש התצוגה עצמה מבוצע על-ידי סטנדרטים בסביבת הפיתוח, כדוגמת JSR-168.	שירותי ממשק המשתמש
5	OGSA-DAI (Data Access & Integration), Grid-FTP, Data Replication, High-level Publish/Subscribe, ניהול האחסון, הגדרת מידע בינרי, ניהול טרנזקציות.	2	יש צורך בפיתוח יעודי על מנת לממש את התהליך בצורה מלאה.	שירותי אחסון המידע
5	קונפיגורציית המשאב/ שירות, ניהול פריסת השירות, ניהול מחזור חיי השירות, Grid Economy Model.	3	תוך התבססות על תקני ה-*WS: WSDM, WS-Management, WS-Transfer	שירותי שו"ב
5	מתבסס על פתרונות *WS.	5	UDDI, WS-Discovery, WS-Resource Framework (WSRF).	שירותי גילוי
5	Virtual Organizations.	1	נושא מורכב מאוד המצריך פיתוח יעודי על מנת לממש את התהליך בצורה מלאה.	שירותי שיתוף
5	תמיכה במערכות Legacy, טרנספורמציית המידע.	4	תוך התבססות על תקני *WS: BPEL, Ws-Choreography, WS-Coordination	שירותי תיווך
44		29		סך כל הניקוד

טבלה 7 - סיכום השוואת התרחישים

10. סיכום ומסקנות

פרק זה מסכם את המסקנות על בסיס תוצאות העבודה. כמו כן, על בסיס מסקנות אלו מפורטים כאן ההמלצות לניצול שירותי ה-Grid למגוון השימושים והמתארים.

10.1. עיקרי המסקנות

כבר היום אפשר לראות כי ההשפעה של ארכיטקטורת השירותים תתרום רבות לדינמיקה העסקית ותוך כדי כך תשפר את היכולת להגיב בצורה יעילה ומהירה יותר לשינויים הנדרשים. השימוש בסטנדרטים יתרום רבות להשגת מטרות אלו, ובעיקר יפחית את זמני הפיתוח של היכולות החדשות. יכולות ה-Grid, אשר הדרישות אליהם מן המערכות הולכות וגוברות, מתואמות עם התפיסה העתידית של מערכות המחשוב הארגוני.

כפי שאפשר לראות, התשתית הקיימת ב-OGSA מקיפה וכוללת חלק ניכר מן ההיבטים התשתיתיים של מערכות ה-IT בארגון, הנדרשים למימוש ה-Grid ESB. בעיקר אפשר לשים לב לתמיכה הרבה בסטנדרטיזציה המובילה בתעשייה והחשיבות שמייחסים לה מפתחי ה-GT4, דבר שיוביל, בסופו של דבר, להפחתה משמעותית בעלויות האינטגרציה של המערכות הקיימות בארגון ואלו שיפותחו בעתיד.

10.2. פירוט המסקנות

פתרון ESB מבוזר, המבוסס על ה-Grid, משלים פערים מהותיים הקיימים ב-ESB, סטנדרטיים מבוססי Web Services ומגשר על הפערים הבאים:

- סטנדרטיזציה – מובילה ל-Interoperability בין השירותים ובכך תורמת לפיתוח יכולות חדשות ולהתבססות על יכולות קיימות, ובעיקר לקישור בין שירותים בין-ארגוניים.
- ביטחון מערכות המידע – הושקעו משאבים רבים בתחום ביטחון מערכות המידע תוך התבססות על הסטנדרטים המובילים כיום (WS-Security ,SAML , X.509).
- Fault Tolerance – שירותים המבוססים על ה-Grid, אשר מטרתם העיקרית לעבוד בסביבה מבוזרת. מספקים את היכולת להתאושש משגיאות, גם של הרשת וגם של השירותים.
- Quality of Service – כיום, כבר יש תמיכה ב-QoS של המשאבים ברשת (כוח העיבוד והזיכרון), כיוון שתכונה זו בלטה במיוחד בפרויקטי Grid המחקריים. עדיין יש להשלים את התמיכה ב-QoS של השירותים (יטופל בעזרת WSDM) ואת התמיכה ב-QoS של הרשת עצמה (כרטיסי הרשת).
- יתרונות ה-Grid – פתרון המבוסס על טכנולוגיית ה-Grid, כאמור, מהווה קפיצת מדרגה ליכולות הקיימות כיום. היתרונות הבולטים במיוחד במערכות התשתית הם:

זמינות גבוהה, יכולת עמידה בנפילות, יכולת גידול גבוהה. קשה לקבל יתרונות אלה ממערכות התשתית הקיימות כיום.

- המשכיות עסקית (Business Continuity) – היכולת הבסיסית של שירותי ה-Grid להתאושש מנפילה של ספק שירות על-ידי חיפוש ספקי השירותים האחרים באמצעות שירותי הספרייה, מהווה קפיצת דרך בסביבות לא יציבות, אשר אין אפשרות להבטיח בהן שספק השירות ו/ או כתובות ה-IP ו/ או שירותי ה-DNS יישארו יציבים.

10.2.1 השוואת הממצאים של Grid Service לעומת אלה של Web Service

כפי שצוין לפני כן, OGSA מגדירה את השירותים המפורסמים כ-Grid Services. יכולתו של Grid Service הוא שירות המבוסס על Web Service ופועל תחת מוסכמות נוספות המגדירות כיצד צורך השירות משתמש בו. המוסכמות נוגעות לנושאים אשר צוינו לפני כן ואין עליהם מענה מספק ב-Web Services.

כפי שראינו בתרחישים, יש פערים משמעותיים ביכולות ה-Web Services. חלק מהם נפתרים במלואן על-ידי ה-Grid Services, לחלק מהם יש פתרון חלקי ואחרים נשארים ללא מענה:

- שמירת ה-State בעבודה עם Web Service: Web Service אינו stateful, כלומר אינו שומר את המצב שבו הוא נמצא ואין לו זיכרון.
  - היתרון בכך הוא פשטות היצירה ותפעול השירות, שאינו דורש זיכרון.
  - החיסרון העיקרי מתבטא בכך שאין אפשרות לבצע תהליך שיחה מורכב/ מתמשך המצריך את שמירת ה-state, שבו נמצא השירות.
  - ב-OGSA יש שימוש ב-WSRF (Web Services Resource Framework), המרחיב את הגדרת ה-Web Service המקורית, המאפשר לשמור את ה-state של השירות על-ידי הגדרת Resource והמספק יכולת לנהל את מחזור החיים של ה-Resource.
- התבססות על XML: ב-Web Services משתמשים ב-XML על מנת להגדיר את השירות (ה-WSDL מבוסס על ה-XML) ולהעביר מסרים בין צרכני השירות וספקיו (המסר כתוב ב-XML).
  - היתרון בשימוש ב-XML הוא הצמידות הנמוכה שלו לקוד. XML מהווה סטנדרט דה-פקטו לייצוג ולהעברת המידע ואפשר להוסיף בעזרתו יכולות חדשות ללא שינוי מימושים של אפליקציות קיימות. נוסף לכך, המידע העובר כ- XML יכול לעבור דרך ה-Firewalls בארגון, כיוון שהקידוד הוא טקסטואלי.

- החסרונות העיקריים בשימוש ב-XML: גודל המידע המועבר כ-XML (יש יחסית הרבה metadata במסמכים) הוא רב וזמן עיבודו גם כן רב יחסית.
- הפתרון המוצע על-ידי OGSA הוא ביזור השירות (סקלביליות), שגורם להפחתת העומס על יחידת עיבוד מסוימת. לה-OGSA אין פתרון לגבי גודל המידע העובר בתווך הרשת, אך אפשר להשתמש בטכנולוגיות לכיווץ המידע העובר בתווך (כיוון ש-XML מבוסס על טקסט, יחס הדחיסה של המידע גבוה יחסית).

- שליחת מסרים בעזרת HTTP: הפרוטוקול הסטנדרטי להעברת המידע ברשת האינטרנט HTTP.
  - היתרון בשליחת מסרים בעזרת HTTP הוא הסטנדרטיות של הפרוטוקול. הפרוטוקול עובד בפורט ידוע (פורט 80) ומותאם לשימוש עם Firewalls.
  - החיסרון בפרוטוקול HTTP הוא חוסר האמינות של המסר הנשלח. לאחר שליחת בקשת HTTP, לא מובטח שהיא תגיעה ליעדה (אין מנגנון מובנה לכך), ואין מעקב מובנה אחר התקדמות המסר.
  - OGSA אינה פותרת את הבעיה, כפי שמרבית הפתרונות עובדים כיום (שימוש ב- Message Brokers על מנת להבטיח שהמסר יגיע ליעדו), אלא מסתמכת על התקן WS-RM (Reliable Messaging) ועל יכולות ניטור השירותים וניהולם. זאת, על מנת להבטיח את זמינות השירות והרשת ובכך את הגעת המסרים.

- טיפול בשגיאות: פעולות מסוימות דורשות הפעלה של יותר משירות אחד ועל המערכת לדעת (איך) להתאושש משגיאות.
  - כיום, אין טיפול בנושא ברמת ה-Web Services.
  - כיוון ש-OGSA בנויה על מנת לשרת מערכות מבוזרות, שיש בהן צורך כדי להתאושש משגיאות, הדגש הושם בטיפול בשגיאות הנובעות מן המשאבים ברשת, בפרט בנוגע לשגיאות הנובעות מזמינות. כיום, אין סטנדרטים המאגדים את הטיפול ב-QoS של הזמינות הרשת, אך יש עבודה בנושא גם ב-OGSA. כרגע, שירותי המידע מטפלים בתמיכה בטרנזקטביות, על מנת להבטיח כי המידע תקין.

- שימוש ב-UDDI: UDDI (Universal Description, Discovery, and Integration) הוא בעצם מאגר מרכזי המכיל מידע על השירותים הקיימים ברשת ואפשר לחשוב עליו כאל "דפי זהב" לשירותים ברשת. ה-UDDI מאפשר לרשום שירותים חדשים ולחפש שירותים קיימים.

- UDDI עוזר לצרכני השירותים למצוא שירותים הנמצאים ברשת ומאפשר לצרכן לצרוך את השירות.
- הבעיה המרכזית ב-UDDI היא היותו מאגר מרכזי יחיד, ובעת נפילתו נוצרת בעיה במציאת השירותים הקיימים ברשת.
- הפתרון המוצע על-ידי OGSA הוא שימוש ב-Index Service מבוזר, כלומר שירות המאפשר לרשום ולחפש שירותים ומשאבים ברשת, שבעזרתו אפשר לשלוט על אופן הסנכרון בין ספקי השירות.

- אבטחת המידע: אחד הנושאים שאינם מטופלים תחת Web Services (במקור). בשנים האחרונות התהוו סטנדרטים בנושא, כגון WSS.
  - OGSA מתבססת על הסטנדרטים הקיימים (במיוחד על WS-Security), אך עדיין אין הנושא שלם לגמרי (בעיקר בנוגע לאכיפת Policy). אבטחת המידע מקבלת דגש רב בפיתוח ארכיטקטורת OGSA ויש עבודה רבה בהגדרת השירותים המספקים זאת.
- Interoperability: כיום, יש בעיה כאשר מנסים לשלב בין Web Services, הכתובים בסביבות שונות (לדוגמה Net ו-Java).
  - הפתרון המוצע ב-OGSA הוא התבססות על תקנים קיימים ובמיוחד על התקנים של ארגון WS-Interoperability, על מנת לאפשר תמיכה מירבית בעתיד (זוהי אחת הסיבות שיש עיכוב בקביעת הסטנדרטים שבהם משתמשים בתחום אבטחת המידע ב-OGSA).

10.3 נושאים למחקר עתידי

כחלק מהעבודה זוהו כמה נושאים הנדרשים לתיקופה של מסגרת מחקר עתידית. בין הנושאים הנ"ל אפשר למנות:

- שילוב עם פתרונות מחשוב הענן (Cloud Computing) – אחת מפרדיגמות המחשוב, אשר מתפתחות כיום באופן מואץ [35]. תפיסה זו דוגלת במתן שירותי המחשוב, לפי דרישה על-ידי אתרי המחשוב ברשת, ומאפשרת לארגונים גמישות רבה, התורמת לניצול ההזדמנויות במקביל להורדת העלויות. אתרי מחשוב אלה יכולים להיות גדולים, כדוגמת אולמות מחשב קלסיים או קטנים, כדוגמת תחנת עבודה ביתית ו/ או מחשב טקטי מוקשח. תפיסת תשתית השירותים המבוזרת, המבוססת על ה-Grid, נושקת לתפיסת מחשוב הענן, והשימוש בה יכול להוות מנוף להרחבה ולמימוש של תפיסת מחשוב הענן בארגון.

- ביצועים – ידוע כי יש ירידה בביצועים כאשר משתמשים ב-Web Services (במיוחד ב-Web Services Security). אפשר לחלק את המסרים העוברים ברשת לכאלה המכילים מידע רב (במיוחד במערכות מידע הנותנות שירותים לגופים האנושיים), ולמסרים קטנים יחסית המפעילים פרוצדורות מרוחקות (RPC). לגבי סוגיה זו יש כרגע כמה פתרונות בתעשייה (במיוחד בתעשיית המכשירים הניידים), אך דרוש מחקר נוסף בנושא על מנת לבדוק את בשלות הפתרונות.
- מערכות Real-Time - אפשר לראות כי תמיכה בשירותים הדורשים Real-Time כמעט לא באה לידי ביטוי, אך אפשר להשלים זאת בעזרת פרויקטים, כגון NaradaBrokering [34], המאפשר העברת מסרים בסביבה מבוזרת, מאובטחת, בתפוקה גבוהה ותחת אילוצים נוספים.
- ניהול השירותים - תקן WSDM הינו תקן חדש יחסית ונמצא בשלבי כתיבה סופיים. התקן תומך ביכולת לנהל את ה-Web Services. המימוש הבא של Globus Toolkit יכלול לבטח תמיכה בתקן זה, שאמור להשלים את התמיכה ב-Quality of Service של השירותים המתפרסמים על ה-Grid.

10.3.1. המלצות לשימוש בשירותים מבוססי Grid

על בסיס המסקנות, ששירותים מבוססי Grid תורמים משמעותית למתארים הקיימים בעולם המחשוב, בפסקאות הבאות נסקור כמה נושאים הראויים להרחבה, אשר יכולים לנצל משמעותית את יתרונות הארכיטקטורה המבוססת על ה-Grid במידע במסגרת מחקר נוסף. נושאים אלה מועלים באמצעות תיאור פתרונות טכנולוגיים המבוססים על ה-Grid לבעיות עסקיות נפוצות.

10.3.1.1. וירטואליזציה

כל מערכת מידע בארגון זקוקה לסט של משאבי מחשוב שונים, כגון: מעבדים (CPU's), שטחי אחסון (Storage), זיכרון פנימי (RAM) ועוד. משאבים אלה נדרשים לעמוד לרשות המערכת על-פי SLA ו-Security Policy נתון.

כיום, שיטת הקצאת המשאבים הללו היא סטטית, כלומר לכל מערכת מוקצים מראש המשאבים הנחוצים לתפקודה, על-פי הערכת העומס המקסימלי הצפוי. החיסרון העיקרי בשיטת הקצאת המשאבים שתוארה היא העובדה כי כמעט בכל תקופת חיי המערכת הממוצעת יש נצילות נמוכה של משאבי הארגון המוקצים לה. נצילות זו טומנת בחובה בזבוז משאבים ועלות כספית גבוהה מן הנדרש עבור הארגון על רכש ותחזוקה.

בעידן ארכיטקטורת Grid המידע, לעומת זאת, שיטת הקצאת המשאבים תהיה דינמית. הקצאת המשאבים הדינמית מתבסס על וירטואליזציה (מיסוך) של משאבי המחשוב בארגון (כפי שתוארו) ומאפשר את שיתוף המשאבים האלה לפי דרישה (On Demand), תוך שימת דגש

על קיום SLA ועמידה ב-Security Policies.

לדוגמה, נניח שקיימת בארגון משימת עיבוד ארוכה הדורשת משאבי עיבוד רבים (חישוב שכר או היתוך מידע עסקי). הקצאת המשאבים למשימה בעידן Grid המידע תפרק את משימת העיבוד ליחידות עבודה מצומצמות ותהיה אחראית על חלוקת יחידות העבודה בין משאבי העיבוד העומדים לרשותה. ניהול יחידות עבודה אטומיות אלו כולל: ניהול העומס, ניטור, טיפול בשגיאות, איסוף תוצאות והיתוכן והחזרת הפלט למשתמש. שיטה זו תאפשר ניצולת גבוהה יותר של משאבי העיבוד בארגון ותחסוך את העלות הכספית של משאבים אלה. דוגמה נוספת היא שטחי האחסון המוקצים למערכות השונות. הקצאת משאבי האחסון בעידן Grid המידע תוכל לספק אחסון לפי דרישה (On Demand). כיוון שיש מיסוד מעל משאבי האחסון, אפשר לשנות את ההקצאה על-פי צרכי המערכות והארגון המשתנים באופן דינמי.

10.3.1.2. סטנדרטיזציה

בעידן ה-SO נעשה שימוש באוסף הסטנדרטים ממשפחת ה*-WS, על מנת להגדיר את האופן שבו מתנהלת האינטרקציה בין צרכני השירותים וספקיהם. אך נמצא פער מסוים בהגדרת ארכיטקטורת ה-ESB, כמו גם פערים מקומיים וחוסר בשלות של חלק מסטנדרטי ה*-WS. לעומת זאת, בעולם Grid המידע יש ארכיטקטורה סטנדרטית, ה-OGSA. את היווצרות הסטנדרט הנ"ל אפשר מתן מענה על הפערים בתקני ה*-WS, כמו גם אימוץ של תקני WS-I להבטחת ה-Interoperability.

10.3.1.3 אינטגרציה ושיתוף באמצעות Virtual Organization

חלק מרכזי בארכיטקטורת Grid המידע הוא השימוש בקהילות שיתוף הידע (VO's), המשמשות להגדרת תיחום המידע והיכולות המשותפות ביניהן וגם את המשתמשים השותפים לקהילה, תחת Security Policies. התמיכה בקונספט הנ"ל מאפשרת את מימוש חזון השיתופיות בין חלקי הארגון השונים לשם מימוש משימות הנתונות באופן יעיל יותר. יכולות שיתוף אלו אינן מהוות באופן בסיסי חלק מההגדרה הכוללת של ESB, אך ניתנות לארגון כחלק מארכיטקטורת OGSA.

10.3.1.4 תמיכה ב-Stateful Web Services

מערכות רבות מורכבות מרכיבי תוכנה (Components), אשר כחלק מהפונקציונליות שלהם הם שומרים על מצבם הלוגי (State). דוגמאות לרכיבים כאלה אפשר למצוא ברכיבי ניהול של ישויות כחלק מתהליך מסוים. יש לציין כי שמירת מצבם הלוגי הוא חלק בלתי נפרד מן הפונקציונליות הזאת.

כיום, Web Services אינם תומכים בקיום של Stateful Web Services, כלומר שמירת ה-State וניהול Session בין צרכן השירות לספק השירות.

ארכיטקטורת ה-OGSA, לעומת זאת, אימצה את תקן WSRF ותומכת באופן מלא בשמירת

ה-State של השירות כחלק מתפיסת הווירטוליזציה, המתארת את האינטרקציה בין הצרכן לבין המשאב הווירטואלי. מכאן נובע שאת יכולות רכיבי התוכנה שומרי המצב הלוגי יהיה אפשר לשתף בצורה סטנדרטית אך ורק בעידן Grid המידע.

10.4. תרומת המחקר

מחקר זה הדגים והוכיח כי ניתן לייצר פלטפורמת שירותים מבוזרת, המאפשרת חשיפה וצריכה דינאמית של שירותים, אשר מקורם יכול להיות אף מחוץ לגבולות הארגון.
המחקר נסמך על הגדרה קיימת של צבא ארה״ב לתשע משפחות השירותים, אשר מימושם מייצר תשתית אפליקטיבית לחשיפה של שירותי התוכנה ברשת הארגונית ולצריכתם.
הגדרה זו ושילובה עסתפיסת ה-SOA וטכנולוגיות Grid מבוזרות אפשרו את יצירתה של תשתית ארכיטקטונית **מבוזרת** – הפועלת ומאפשרת שיתוף ללא מרכז פיזי אחד כפי שהוכח בפרק 9.1.8, **גמישה** – התומכת בגילוי דינאמי של שירותים כפי שהודגם בפרק 9.1.7 ופורט בפרק 9.2.4, **שרידה ומנוהלת** - כפי שהוכח בפרק 9.1.6, שביכולתה לאפשר חשיפה וצריכה דינאמית של שירותים ממקורות שונים בארגון ומחוצה לו.

הארכיטקטורה החדשה מאשרת, בין השאר, את יכולת **הזמינות והסקלאביליות** של התשתית המאפשרת גידול בצורה פשוטה ויעילה כפי שהודגם בפרק 9.2.3, תוך תמיכה במספר רב של משתמשים ושירותים, ואת נושא **אבטחת המידע** והשירותים הנדרשת מארכיטקטורה מבוזרת כפי שהודגם בפרק 9.1.1 והורחב בפרק 9.2.2.
ליכולות אלו יתרונות ברורות על פני ארכיטקטורות אחרות כפי, שפורט בפרק 10.2.1.
החיזוק ליכולות הארכיטקטורה וחדשנותה הוסקה באמצעות עריכת השוואה מקיפה עליה אל מול זו הקיימת היום לצריכה ולחשיפה של שירותים (ארכיטקטורת Enterprise Service Bus).
ההשוואה, המבוססת על מודל ה-Architecture Tradeoff Analysis Model (ATAM), מובילה להשוואה איכותית בין פרמטרים שונים המייצגים את יכולות תשע משפחות השירותים המאפיינים ארכיטקטורות אלו. הן הקנו יתרון לארכיטקטורה החדשה והמוצעת, בייחוד ביכולות הגידול, הגמישות והזמינות של שירותי התשתית המרכיבים את הארכיטקטורה, והמאפשרים את יכולותיה הדינמיות, כפי שהוצג בפרק 9.3.

ארכיטקטורה זו הינה חלק מגל מחשובי, אשר לאחרונה אנו חווים את תחילתו תחת הכותרת של שירותי מחשוב מבוסס ענן (Cloud Computing). שירותי מחשוב מבוססי ענן הם, למעשה, הדגמה חיה של תפיסת ה-Grid, החוצים את הגבולות החיצוניים של הארגון והמאפשרים צריכה דינמית של היכולות דרך רשת האינטרנט.
מחשוב מבוסס ענן אינו יכול להתקיים לאורך זמן ללא תמיכה יסודית ביכולות אלו, ובעיקר של אבטחת מידע, גילוי ושיתוף דינאמי של שירותים ויכולת גידול מהירה. דרך תרחישי השימוש, שהוצגו בפרק 8.10 והודגמו בפרק 9, הוכחנו שתשעת משפחות שירותי הבסיס ישימים בצורה ברורה. מכאן, שמימוש תשע משפחות השירותים באמצעות ארכיטקטורה חדשנית זו מקנה תשתית לרשת שירותים ההולכת בהלימה לתפיסת הענן.
הארכיטקטורה והתשתית המוצעים במחקר, יחד עם המלצות שהורחבו בפרק 10.3.1 בתחומי

הוירטואליזציה, הסטנדרטיזציה והשיתוף, מהווים נדבך חשוב ומקיף להמשך התפתחותו של מחשוב בתפיסת הענן. התשתית הארכיטקטונית המוצעת מקדמת את יכולות המחשוב שצרכני השירותים וספקיהם נדרשים אליהם, על מנת לספק שירותים אפליקטיביים בעידן הענן.

11. רשימה ביבליוגרפית

[1] Papazoglou P. Michael: *"Extending the Service Oriented Architecture"* - Business Integration Magazine, pp. 19-21, February 2005

[2] Ian Foster, Carl Kesselman, Steven Tuecke, J.M. Nick : "*Grid Services for Distributed System Integration"* - IEEE Computer, 35(6), 2002

[3] Willaim K. Cheung et all. : *"Towards Autonomous Service Composition in A Grid Environment"* – School of Computer Science & Engineering, University of New South Wales, Australia & Baptist University of Hong-Kong, 2004

[4] Ian Foster, Steven Tuecke: *"Describing the Elephant: The Different Faces of IT as Service"* – Enterprise Distributed Computing Magazine, July/August 2005

[5] IEEE Internet Computing, Special Issue: *Middleware for Web Services*, 2003

[6] IEEE Process Computing, Special Issue: *Grid Computing*, Edited by M. Parashar and C. A. Lee. Vol 93, No 3, March 2005

[7] Bass Tom & Mabry Roy: *"Enterprise Architecture Reference Models: A Shared Vision for Service-Oriented Architectures"* – Submitted to IEEE MILCOM 2004

[8] Pitoura E., Abiteboel S., Pfoser D., Samaras G., Vazirgiannis M.: *"DBGlobe: A Service-Oriented P2P System for Global Computing"*- University of Ioannina Greece 2003

[9] The UDDI project www.uddi.org

[10] The Globus Alliance www.globus.org

[11] The global Grid forum www.ggf.org

[12] Stevens Michael: "*the benefits of Service-Oriented Architecture*", www.developer.com/design/article.pho/1041191

[13] Webber Jim, Wataon Paul, Parastatidis Savas: "*Using Web Services to Build Grid Applications – The No Risk WSGAF Profile*" - School of Computing Science, University of Newcastle upon Tyne, June 2004

[14] Ian Foster, Carl Kesselman, Steven Tuecke : "*The Anatomy of the Grid"* http://www.globus.org/research/papers/anatomy.pdf

[15] Ian Foster, H. Kishimoto, A. Savva, D. Berry, A. Djaoui, A. Grimshaw, B. Horn, F. Maciel, F. Siebenlist, R. Subramaniam, J. Treadwell, J. Von Reich: "*Open Grid Services Architecture (OGSA)",* http://forge.gridforum.org/projects/ogsa-wg ,2005

[16] Michelson Brenda : "*Enterprise Service Bus Evaluation Framework*" – Patricia Seybold Group, July 2005

[17] F. Leymann: "*Web Services: Distributed Applications without Limits*" – Database systems for Business, Technology and Web, 2003

[18] Fiorano Software, White Paper : *"Dynamic Web Services*", 2003

[19] IBM – *"living in an on demand world"*, http://www-3.ibm.com/e-business/doc/content/feature/offers/whitepaper.pdf ,2002

[20] IBM Red Book – *"Patterns: SOA with Enterprise Service Bus*", 2005

[21] IBM Red Book – "*Grid Services Programming and Application Enablement*", 2004

[22] Vietmeyer Rob, Raines Geoff : "*Global Information Grid – Core Enterprise Services Strategy*" - Network and Information Integrating /DoD Chief Information Officer, September 2003

[23] Alberts S. David, Garstka J. John, Stein P.Fredrick: "*Network Centric Warfare*" – DOD/CCRP 2nd edition 1999

[24] The CBDI Forum: "*Time to Board the Enterprise Service Bus?"*, http://www.cbdiforum.com/secure/interact/2004-07/Enterprise_Service_Bus.php

[25] Chess M. David, Kephart O. Jeffrey : "The Vision of Autonomic Computing", IBM Thomas J. Watson Research Center, IEEE Computer Society, Jan 2003

[26] The CBDI Forum: "*Communicating SOA"*, http://www.cbdiforum.com/public/events/workshops/Communicating_SOA.php

[27] A. T. Mannes, J. Lewis: "*Turning the Network into the Computer: The Emerging Network Application Platform*"- Burton Group, 2004

[28] J. Bloomberg, R. Schmelzer: "*The "Pros and Cons" of Web Services*"- Zapthink Foundation Report, 2003

[29] Cao Junwei: *"Agile Computing on Business Grids"* – C&C Research Laboratories NEC Europe Ltd., June 2003

[30] Michael R. Werder: "*Pricing in the Service-Oriented IT World*" – University of Zurich, January 2004

[31] N. Cook: "*Network Roles*" – Jane's Defense Magazine, August 2005

[32] Ken Birman, Robert Hillman, Stefan Pleisch, “Building Net-Centric Military Applications over Service Oriented Architectures”, SPIE conference, March 2005

[33] UCM model:
http://jucmnav.softwareengineering.ca/twiki/bin/view/UCM/UCMNavDemo

[34] Narada project: http://www.naradabrokering.org

[35] Grid Computing Environments Workshop, 2008. GCE '08, "Cloud Computing and Grid Computing 360-Degree Compared":
http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4738445

[36] Rick Kazman, Mark Klein, Paul Clements: “*ATAM: Method for Architecture Evaluation”,* Carnegie Mellon University, 2000

[37] Teun Lucassen, Iwan de Kok: *“Using Sectors in a Multi Agent Approach to a Taxi Planning Problem”,* University of Twente, 2006

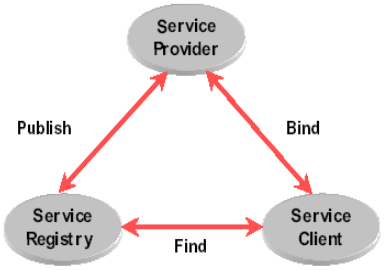
[38] Jiang, C. J., et. al.: “*Urban Traffic Information Service Application Grid*”, Journal of Computer Science & Technology, Vol. 20, No. 1, 2005.

נספח א – סקירת פרדיגמת ה-SOA (Service Oriented Architecture)

התפתחות עולם האינטרנט ורשת ה- Web הביאה עימה את תפיסת השירותים האפליקטיביים ברשת (Web Services), שיצרה הזדמנויות עסקיות וטכנולוגיות חדשות והביאה חסכון בפיתוח כפול ומיותר של אותן אפליקציות.

שירותים אלו שפעלו ברשת אפשרו פשטות, אחידות, חסכון ודינאמיות רבה יותר בצריכה של השירותים. שירותים אלו יצרו מהפכה באופן החשיבה והפיתוח של מוצרים ואפליקציות חדשות, ובאו לידי ביטוי במהרה ברשת העולמית והארגונית. כך, הבשילה התפיסה כי ניתן להתבסס על הרשת כמקור ארגוני משותף לצריכת שירותים, ואף יותר מכך – כמקור המאפשר חשיפה וצריכה **אוטונומית** של השירותים.

אבולוציה זו הביאה להבנה שניתן לתפוס את הרשת כ-"ארכיטקטורה מכוונת שירותים" – Service SOA – Oriented Architecture כרעיון מסדר וחסכוני לפיתוח האפליקטיבי הארגוני. העיקרון הבסיסי עליה מושתת הבנה זו היא שכל שירות יכול לחשוף עצמו ברשת, להירשם בקטלוג משותף (Publish), להיחשף לשירותים אחרים (Find) ולצרוך אותם בצורה אוטונומית (Bind) ומנוהלת[1].



איור36 - ארכיטקטורת שירותים בסיסית [1]

למעשה, פרדיגמת ה-SOA מציגה תפיסה חדשה לפיה ניתן להפסיק עם הפיתוח הוורטיקאלי של מערכות מונוליתיות (Monolithic) שלמות, ולהחליפן בצריכה מודולארית של מספר שירותים על גבי הרשת. תודות לכך מתאפשר ביצוע איחוד (Aggregation) של מספר שירותים לשירות חדש אד-הוקי לטובת משימה מוגדרת (CA – Composite Application)[3]. בצורה זו הפרדיגמה מעבירה את כובד המשקל מהמערכות הארגוניות (System Centric) לרשת הארגונית (Net Centric) ומעלה שאלות מחקריות רבות ביחס לשירותים הבסיסים אליהם נדרשת הרשת[7].

תפיסת השירותים כפי שנגזרת מהגדרת ה- SOA אינה תפיסה חדשה, אלא פרדיגמה הקיימת מזה מספר שנים המנסה ליצור סדר ושימוש חוזר (Reuse) ביכולות קיימות הנחשפות כשירותים (Services).

כיצד התבצעה האבולוציה של SOA

כבר לפני שנים בנו מומחי ה-IT וישמו בהצלחה אפליקציות מבוססות SOA לפני ש–XML ו-Web Services הפכו ללהיט הם רק דיברו על תהליכים תוך שימוש במונחים כמו מדולריות, רכיבים הניתנים לשימוש חוזר, object-oriented-programming או application programming interfaces. למרות שאף אחד מהקונספטים הללו אינו זהה ל SOA, הם כולם מכילים אספקטים שלו [17].

בטרם נצלול פנימה לתוך עולם ה SOA, הבה נביט אחורה על האבולוציה שלה. בכדי לעשות זאת עלינו פשוט להתבונן באתגרים שעמדו בפני המפתחים בעשורים האחרונים, ולראות את הפתרונות שעלו בכדי לפתור בעיות אלו.

מפתחים מוקדמים הבינו שכתיבת תוכנה הופכת מורכבת יותר ויותר. הם היו צריכים דרך טובה יותר לעשות שימוש חוזר בקוד שהם עצמם כתבו. כאשר חוקרים הבחינו בכך, הם הציגו את התפישה של תכנון מודולארי. בעזרת עקרונות התכנון המודולארי, הצליחו המפתחים לכתוב פונקציות ותחליפים ולבצע שימוש חוזר לקוד. זה היה נפלא למשך תקופה קצרה. מאוחר יותר המפתחים החלו לראות שהם חותכים ומדביקים את המודולים שלהם לצורך אפליקציות אחרות, וזה החל ליצור בעיות חריפות של תחזוקה; כאשר נתגלה באג בפונקציה כלשהיא, הם נאלצו למצוא את כל האפליקציות שהשתמשו בפונקציה ולשנות את הקוד בהתאם. לאחר פתרון התסבוכת הזו, החל הסיוט של ה deployment. המפתחים לא אהבו זאת, ונזקקו לרמה גבוהה יותר של הפשטה.

זו הייתה שעתה של תפיסת ה- object-oriented בכדי לפתור זאת, אך יחד עם זאת עלו גם הרבה בעיות אחרות. ככל שגדלה מורכבות התוכנה, החלו המפתחים לראות שישום ותחזוקה של תוכנה הוא מורכב, והם רצו דרך לבצע שימוש חוזר ולשמור פונקציונאליות, ולא רק קוד. החוקרים הציעו עוד שכבה של הפשטה כדי להתמודד עם מורכבות זו על ידי component-based software שיטה זו הייתה ועודה פתרון טוב לשימוש חוזר ולתחזוקה, אולם לא טיפלה בכל המורכבות מולה ניצבים מפתחים כיום כמו תשתיות שונות רשת האינטרנט ועוד. מול כך עולה שארכיטקטורה מבוססת שירותים (יחד עם Web Services) מספקת פתרון לבעיות אלו. על ידי אימוץ של SOA מצמצמים את התלות והבעייתיות של פרוטוקולים ופלטפורמות, ואינטגרציה מתבצעת בצורה פשוטה ושקופה.

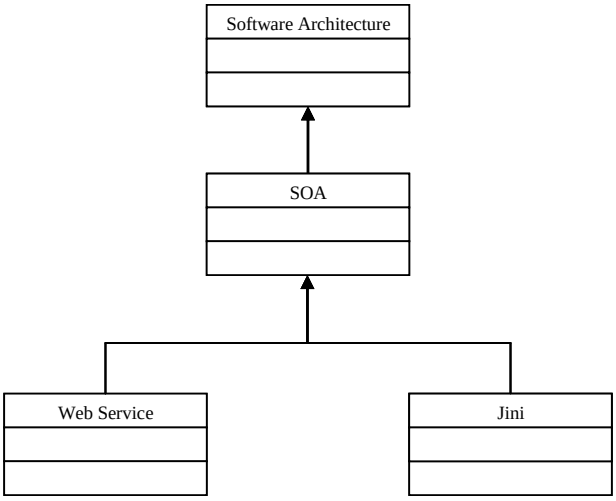
הסבר תהליכי לפרדיגמת ה SOA

פרדיגמת ה- SOA מושתת על ארכיטקטורה הבנויה מרכיבים רבים ובעלת יכולת התקשרות רבה ללא מגבלה כלשהי של פלטפורמות שונות, שפות תכנות שונות, ישום ועוד, תודות לשקיפות של מיקום השירותים.

הארכיטקטורה הארגונית מגדירה את הרכיבים וכיצד קשרי הגומלין פועלים ברמה הגבוהה. כאשר קיימים כבר רכיבים שונים שפועלים במערכות הארגוניות אנו צריכים דרך להשתמש בהם יחד, אולם קיימת הבעיה שהמפתחים של הרכיבים בונים אותם בצורות שונות. למשל על ידי שימוש ב- EJB , CORBA או כל מיני דרכים שונות בהתאם למערכת עליה אנו עובדים . אומנם המפתחים משתדלים שניתן יהיה לקשר בין רכיבים מסוימים אולם ישנם כאלו שממש קשה להתקשר אליהם כיוון שהם עובדים על פלטפורמות שונות או מתודולוגיות שונות. גם אם לא הייתה קיימת בעיה זו, עדיין רכיבים צריכים לדעת יותר פרטים על הרכיבים אליהם הם רוצים להתקשר כדי ליצור קשר שיעבוד. למשל צריך

לדעת את הנתיב, את סוג ההתקשרות ,סוגי הנתונים שיש להעביר וכן בדיוק היכן נמצא הרכיב או השרות, ואיך להתקשר לממשק, ואז אם משתנה הרכיב אליו אני מתקשר אזי הרכיב שלי צריך להשתנות בהתאם.

תפיסת ה-SOA נותנת תשובה לבעיות הללו. התמונה הבאה באה להראות כיצד טכנולוגיות שונות יכולות ליישם את SOA . Web Services הם אחד ממספר טכנולוגיות שיכולות ליישם זאת באופן מוצלח.

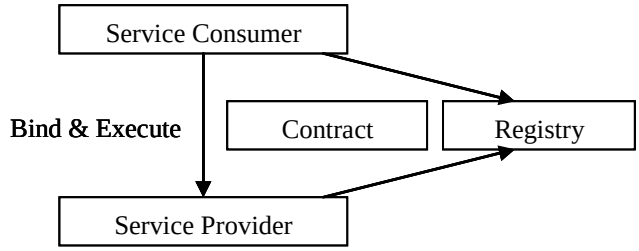


איור37 – אפשרויות מימוש לארכיטקטורת ה-SOA

ההיבט החשוב ביותר ב- SOA הוא ההפרדה בין יישום השירות לבין הממשק עצמו. במילים אחרות הוא מבדיל בין ה-מה ל-איך. הצרכן המשתמש בשירות רואה את השירות פשוט כמוצר שתומך בבקשה ספציפית שלו. לא מעניין את הצרכן כיצד השירות מוציא לפועל את הבקשה מה שמעניין אותו הוא שהשירות אכן יבצע את הבקשה. כמו כן הצרכן מצפה שהשירות יבוצע בהתאם להסכם שהוגדר על ידי שני הצדדים , הדרך בה מבצע השירות את הפעולה בעצם לא מעניין את הצרכן. השירות יכול להתבצע על ידי מימוש Servlet ,אפליקציית VB או כל צורה אחרת , הדרישה היחידה היא שהשרות שולח ומקבל בקשה לפי הפורמט שהסכם על ידי הלקוח.

מרכיבי ארכיטקטורת SOA

שירות ה- "find bind & execute" מאפשר לצרכן של שרות מסוים לבקש אותו מה-registry של צד שלישי. אם אכן קיים ב-registry כזה שירות, הוא נותן ללקוח אפשרות וכתובת להפעיל את השירות על-פי חוזה שרות מוגדר[24]:



איור38 - מרכיבי ארכיטקטורת ה-SOA

SOA תומך ב- "find bind & execute" על ידי הישויות הבאות:

- Service Consumer

ה- Service Consumer הוא אפליקציה, שירות או כל סוג אחר של תוכנה שדורשת שירות כלשהו. זו הישות שיוזמת את מציאת השרות ברישום (registry) יצירת הקשר עם השרות והוצאה לפועל של הפונקציות של השרות המבוקש. ביצירת הקשר מוסכם בעצם על דרך ההתקשרות בין מבקש השרות לבין השרות עצמו. הלקוח מפעיל את הבקשה על ידי שליחה של בקשה שבנויה בדיוק לפי ההסכם אותו קיבלו שני הצדדים.

- Service Provider

ה- Service Provider הוא השרות, הוא הישות המבוקשת שמקבלת ומוציאה לפועל את הבקשות של הלקוח. היא יכולה להיות מערכת main frame, רכיב כלשהו או כל מערכת שמוציאה לפועל בקשות לשירותים שונים. ה- Service Provider מפרסם את הסכם ההתקשרות ב-registry על מנת לאפשר ל-Service Consumerגישה לשרות.

- Service Registry

ה- Service Registry הוא תיקיה על הרשת בה יש את כל השירותים הזמינים. היא ישות המקבלת ושומרת את הסכמי ההתקשרות מ- Service Provider ומספקת את ההסכמים ל-Service Consumer המעוניינים בשירות [9].

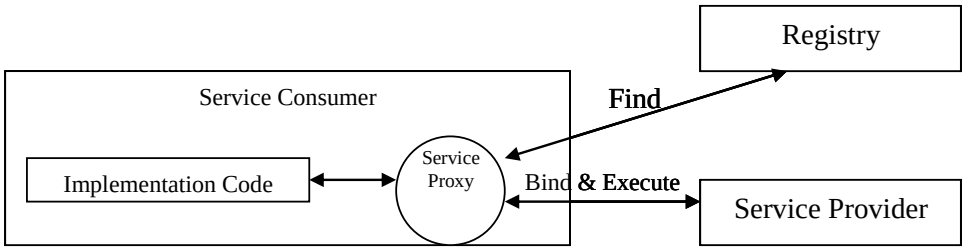
- Service Contract

ה- Contract הוא מפרט של הדרך בה לקוח של שירות מסוים יתקשר עם הספק של השרות. הוא מפרט את הפורמט של הבקשה מהשרות והתשובה ממנו. כל Service Contract יכול לדרוש תנאים מוקדמים ומאוחרים. דרישות אלו מגדירים את המצב בו צריך להיות השירות על מנת לבצע פונקציה מסוימת. הוא גם יכול לדרוש רמת איכות שרות מסוימת (QoS). רמות QoS הם דרישות להיבטים לא פונקציונאליים של השרות.לדוגמא תכונת איכות של שרות הוא הזמן שלוקח להפעיל פונקציה של שרות כל שהוא.

- Service Proxy

ה-Service Provider מספק Service Proxy ל-Service Consumer. ה-Service Consumer מפעיל בקשה על ידי קריאה לפונקצית API על ה-Proxy. כפי שמתואר בתמונה הבאה ה-Service Proxy מוצא מפרט מוסכם ומפנה ל- Service Provider ב- registry. לאחר מכן הוא מעצב את הבקשה בהתאם

למפרט ומפעיל את השירות בשם הלקוח. ה- Service Proxy הוא ישות נוחה לשימוש ה- Service Consumer. הוא לא דורש מהמפתח של ה- Service Consumer לכתוב תוכנה מותאמת לגישה ישירה לשירות.



איור39 - אופן פעולת ה Proxy בארכיטקטורת השירותים

ה- Service Proxy יכול להגביר את הביצועים על ידי כך שהוא זוכר (ב- cache) הפניות ונתונים שונים. כאשר ה Proxy עושה caching להפניה מרוחקת קריאות עוקבות לשירותים לא ידרשו קריאות שונות ל- registry. על ידי אגירה של מפרטים של שירותים מקומית, הלקוח מפחית את מספר הבקשות ברשת לביצוע השרות.

בנוסף ה Proxy יכול לשפר ביצועים על ידי הורדה של קריאות ברשת בכך שהוא יבצע פונקציות מסוימות באופן מקומי. לשירותים מסוימים שלא צורכים שרותי נתונים כל הפונקציה יכולה להיעשות באופן מקומי ב-Proxy. למשל, פונקציות כמו חישובים מתמטיים שונים יכולים להתבצע ב-Proxy, או פונקציה שצריכה שרות נתונים קטן יחסית, ה-Proxy יכול להוריד את הנתונים פעם אחת ולהשתמש בהם במקרה הצורך. העובדה שהפונקציה מבוצעת ב-Proxy ולא נשלחת לבצוע בשרות מרוחק היא שקופה ל- Service Consumer. אולם כאשר משתמשים בטכניקה הזו מאוד חשוב שה Proxy יתמוך רק במתודות שהשרות עצמו מספק. ההגדרה של ה Proxy היא שהוא פשוט הפניה לוקאלית לאובייקט מרוחק. אם ה Proxy משנה בדרך כל שהיא את הממשק של השרות המרוחק אז טכנית הוא בעצם כבר לא Proxy.

ה-Service Provider מספק Proxy לכל סביבה. ה- Service Proxy נכתב בשפה המקורית בה נכתב ה- Service Consumer , למשל Service Provider יכול לתאר Proxy's שונים ל- java, Delphi, VB אם אלו הפלטפורמות ל- Service Consumer. למרות שה- Service Proxy לא נדרש הוא יכול לשפר באופן משמעותי את הביצועים ונוחות ל- Service Consumer.

- Service Lease

ה- Service Lease אותם נותן ה- registry ל- Service Consumer מפרט את הזמן בו החוזה בין המתקשרים בר תוקף, רק מהנקודה בו הלקוח מבקש חוזה מה- registry עד לזמן שהוגדר ב- Lease . כאשר נגמר החוזה, הלקוח צריך לבקש חוזה חדש מה- registry.

החוזה נחוץ לשירותים שצריכים לשמר מידע על המצב הנוכחי בין הלקוח לנותן השרות. ה-חוזה מגדיר את הזמן בו המצב יכול להישמר כמו שהוא. בנוסף הוא גם מצמיד בין Service Consumer ו- Service

Provider על ידי הגבלה של הזמן בו הלקוח וספק השרות יהיו קשורים. ללא חוזה כזה לקוח יכול להיות מקושר לשרות תמיד, ולא יוכל להתקשר לחוזה חדש עם שרות אחר. בעזרת החוזה אם יש פרוצדורה שצריכה לשנות את היישום שלה, היא יכולה לעשות זאת רק כאשר החוזה עם ה- Service Consumerהסתיים. היישום יכול להשתנות ללא השפעה על התהליך שכרגע מתבצע ב- Service Consumer בגלל שהלקוחות שכרגע משתמשות בשרות יצטרכו לבקש חוזה חדש וכאשר יקבלו את החוזה החדש הם ישתמשו ביישומים החדשים ששונו קודם.

מאפייני SOA

לפרדיגמת ה- SOA מתלווים המאפיינים הבאים [1,12]:

- שירותים חייבים להיות גלויים ודינאמיים.
- שירותים פועלים באופן עצמאי ומדולרי.
- שירותים צריכים להיות מעוצבים כך שיוכלו לקיים שיחה בינם לבין עצמם.
- שירותים צריכים להיות מעוצבים כך שיוכלו להתחבר בחופשיות.
- שירותים צריכים להיות בעלי ממשק שניתן לגישה ברשת.
- ממשק רחב ככל האפשר (ניתן לגשת לשרות מסוגים שונים של אפליקציות ,מערכות שבנויות בטכנולוגיות שונות).
- מיקום השירותים צריך להיות ברור לחלוטין.
- אפשרות של התאוששות במצבים שונים.

היתרונות של אימוץ SOA

1. SOA הופכת את הצורך לבצע אינטגרציה של סביבת ה-IT בארגונים לקלה יותר. אחד הערכים הגדולים של SOA היא יכולתה לעבוד היטב בסביבות הטרוגניות כאשר מפתחים אינם נדרשים לבזבז כמויות גדולות של זמן בכתיבת שורות קוד חדשות בכדי לקשר בין אפליקציות. במקום זאת הם יכולים להשתמש בפרוטוקולים סטנדרטיים, כגון web-services. חתיכות גדולות של קוד ה SOA ניתנות לשימוש חוזר, ומפחיתות עליות של פיתוח. SOA לוקחת את השקעות המורשת כגון ה-SAP ו-Siebel, אורקל ו-DB2, ומאפשרת להם לפעול יחד יפה ובעלות נמוכה.
2. כוחו של ה-SOA הוא במינוף הקיים. לא נדרש לשנות או להחליף את המערכות הקיימות במערכות חדשות אלא להחצינן כשירותים ברשת. על ידי זיהוי היכולות של המערכות הקיימות ניתן למקסם את הערך של ההשקעות ב –IT תוך כדי מזעור הסיכון והפיכתו למינימאלי. כמו כן בניית שירותים תוך שימוש ב SOAP (simple-object-access-protocol) וב-WSDL, לא רק שמקילה על תהליך האינטגרציה הפנימית, היא גם מאפשרת ללקוחות ושותפים עסקיים לחלוק אינפורמציה בקלות רבה יותר בין Firewalls של החברות.

3. יתרון נוסף של ה-SOA היא היכולת להוביל לדיאלוג טוב יותר בין מנהלי המחשוב הארגוני לבין מנהלי העסקים בארגון על ידי הבאת עובדי ה IT לחשוב במושגים של ארכיטקטורת עסקים, ולא של טכנולוגיה.
4. חסכון בעלויות הפיתוח - השירותים שנבנו על ידי המפתחים יתפתחו לקטלוג של שירותים שניתנים ל- reuse. מפתחים יגיעו להבנה שהקטלוג הזה הוא סט של נכסים שניתנים להנצלה ושימוש חוזר, ויכולים להשתלב בדרכים שלא חשבו אליהם קודם לכן. הארגון כולו ירוויח מאפליקציות חדשות שמפותחות מהר יותר כתוצאה מהקטלוג הזה.
5. SOA מחייבת את המפתחים לתכנן אפליקציות כאוסף של שירותים, גם אם לא נראה שיש יתרון מיידי ובולט בעשייה זו. ארכיטקטורה מבוססת שירותים מחייבת את המפתחים לחשוב מעבר לאפליקציה הנוכחית שלהם, לשקול אם להשתמש שוב בשירותים הקיימים ולבחון כיצד מפתחים אחרים עשויים להשתמש שימוש חוזר בשירותים שהם יוצרים. סוג כזה של מבנה אפליקציות מאפשר לחברה להגיב במהירות לתנאי השוק המשתנים.

נספח ב – סקירת נושא ה-ESB [2] (Enterprise Service Bus)

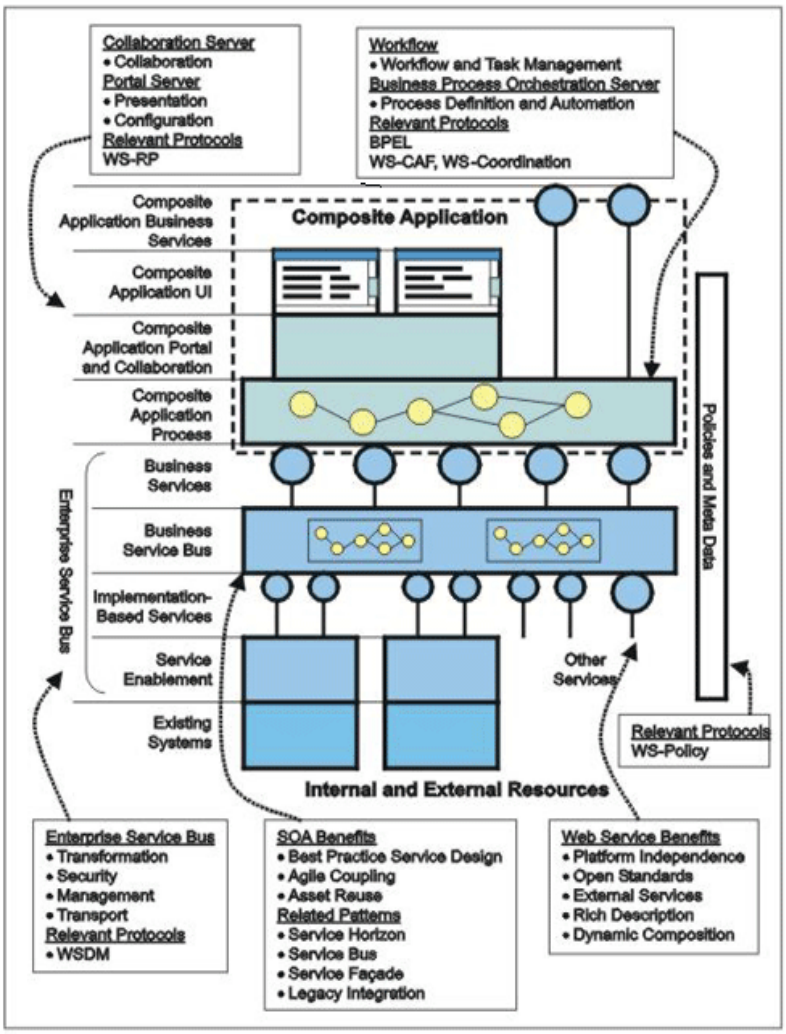
ה ESB הנו תשתית תיאום ואינטגרציה ארגונית אשר אמורה לאפשר אינטראקציות מנוהלות ומאובטחות בין צרכני וספקי שירותים בעלי סביבה טכנולוגית וסמנטיקה שונות. ה- ESB למעשה מהווה מימוש של פרדיגמת ה- SOA, לרוב דרך שימוש ב Web Services. ניתן לראות את ה ESB כמימוש של Bus ארגוני עליו מפרסמים את השירותים של הארגון, ודרכו מאפשרים את צריכתם הדינאמית:

איור 40 - סכימה כללית למבנה ESB [24]

נכון להיום לא קיימת תמימות דעים לגבי הגדרה ברורה וקוהרנטית של מושג ה ESB או לגבי תיחומם של התכנים והיכולות תחת מושג זה. חלק מהגדרות רואות בתשתיות תזמור שירותים (Orchestration) חלק מארכיטקטורת ה ESB, ואילו אחרות לא. על אותו משקל ישנם חברות בתעשייה וספקים אשר אורזים לתוך חבילת ה ESB שלהם תשתיות/מוצרי MOM ו EAI. על מנת לעגן את ההגדרה של ESB נאמץ את ההגדרה הרשמית על-פי ה CBDI כבסיס לראייה והבנה של קונספט ה-ESB: "שכבת ה Enterprise Service Bus (ESB) מהווה תבנית ארכיטקטורה לוגית המספקת מבט (Infrastructure View) ליכולות התשתית הבסיסיות של הארגון. שכבת ה ESB עושה זאת תוך הדגשה של שכבת התווכה בארגון (Middleware Layer)" [24].

שכבה זו מספקת את היכולות הנדרשות למימוש של תשתית הריצה והניהול על ידי ביצוע Plug In של שירותי התשתית אשר רלוונטיים לארגון (ISB), ומשמשת את מגוון השירותים בארכיטקטורת SOA (גם שירותים עסקיים וגם שירותי תשתית) כפי שניתן לראות בהרחבה באיור 41:

[2] מבוסס על מחקר ופרסום של קבוצת CBDI Forum בתחום[26]

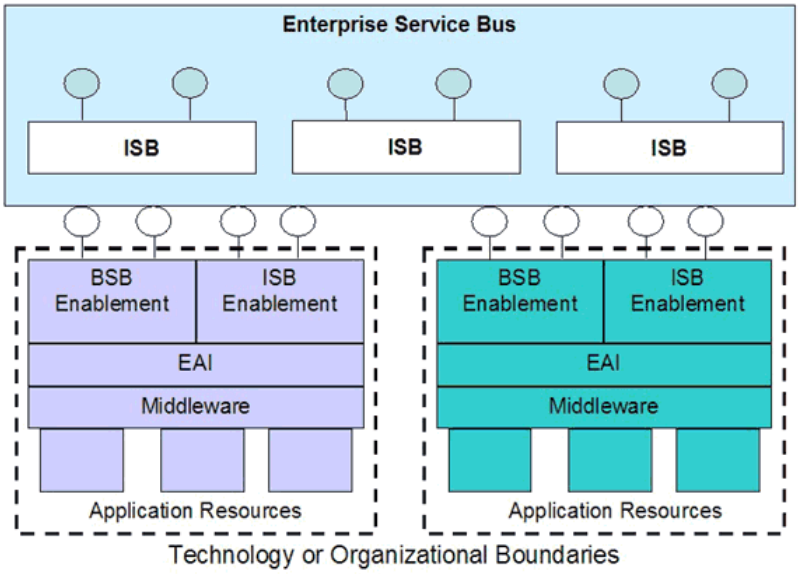


איור 41 - ארכיטקטורת Service Oriented Business Application [24]

שירותי ה ESB הנם שירותי תשתית רוחביים המשמשים משמשים את כלל השכבות בארכיטקטורה. במימוש שכבת ה ESB ניתן דגש מרכזי ליכולת לשלב את יכולות התשתית הקיימות בארגון תחת תשתית אחת, תשתית ה - ESB.

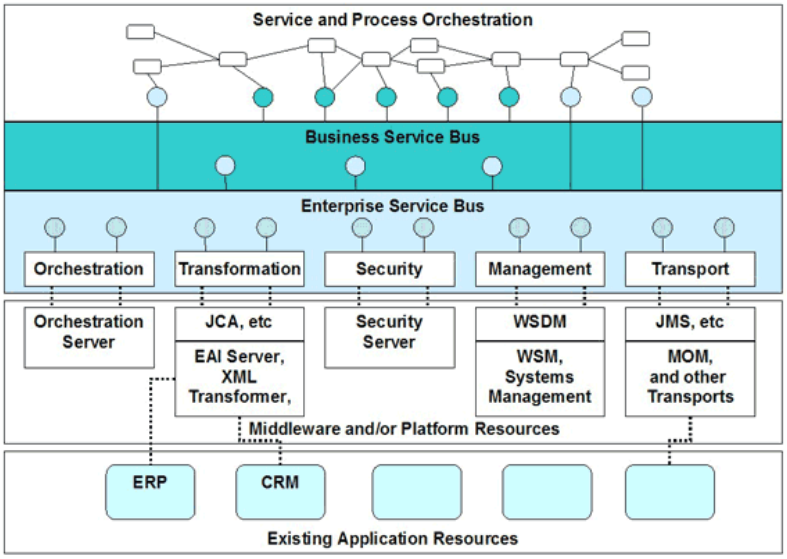
על-פי CBDI ניתן לראות ב ESB שכבה ארכיטקטונית, המאגדת אוסף של שירותי תשתית תוך יצירת שכבת אבסטרקציה לוגית בין שירותי התשתית למימוש הפיזי על ידי מוצרי תוכנה ותווכה רלוונטיים. למעשה כל אותם יתרונות הנובעים מהצמוד הרופף בין השירות למימוש של השירות כפי שבאים לידי ביטוי ב BSB, קיימים גם כאן בשכבת ה תשתיות ובארכיטקטורת ה ESB.

ניתן לראות בשכבת ה ESB כשכבה אשר משלימה את תשתיות ה Middleware (MOM ו EAI לדוגמא) הנפוצות בארגון. למעשה שכבת ה ESB ממנפת ועושה שימוש מאורגן ומושכל ביכולות אלו, כל זאת תוך שמירה על צמוד רופף והפרדה בין שכבת שירותי התשתית לשכבת המימוש הפיזי של התשתיות, דבר המאפשר גמישות רבה בהרחבה, החלפה, ושדרוג של יכולות התשתית השונות.



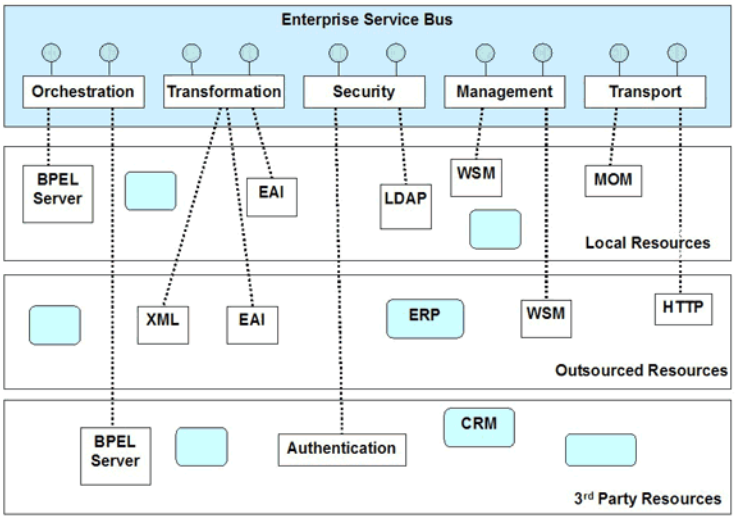
איור 42 - שיתוף עם פתרונות קיימים (ESB Co-Exists with Existing Middleware) [24]

תשתית ה ESB מספקת שכבת שכבה של אבסטרקציה ווירטואליזציה מבוססת SOA לצורכי גישה לאוסף של יכולות תשתית בארגון דוגמת: Transport, Management, Security , Transformation, Orchestration .



איור 43 - שכבות רשת השירותים [24]

מודל ה ESB מורכב מאוסף של ISB (Infrastructure Service Bus) עבור ה - Infrastructure Domains השונים. דוגמאות ל ISB הנם: Security Domain ISB , Communication Domain ISB, Mediation Domain ISB, וכדומה.



איור 44 - וירטואליזציה של משאבים תשתיתיים ועסקיים [24]

באופן כללי ניתן לומר כי ה ESB מהווה גורם מאפשר מרכזי (Enabler) למימוש של ארכיטקטורה SOA בארגון, אם SOA הוא הרעיון או הארכיטקטורה אזי ESB הוא אוסף יכולות התשתית המאפשרות לממש את הרעיונות הגלומים ב SOA.

על מנת לממש פתרון ESB ארגוני נדרש לעמוד בשני יעדים[4] :

- הקמת סט שירותי תשתית אשר יהיו זמינים דרך ה – ESB ויאפשרו סט שלם של אפליקציות חדשות. סט זה של שירותי תשתית צריך לכלול ברמה העליונה: שירותי תצוגה, ניהול שיחה, גישה לבסיסי נתונים, ניהול אירועים וניהול תהליכים המבוססים על שירותי ניהול ברמת הביניים (עמידה במדיניות כגון ביצוע Audit, ניהול טרנזקציות, ניהול זהויות וניהול תקשורת). שירותי הניהול ברמת הביניים מבוססים על שכבה בסיסית של שירותי תשתית המוכרים לנו היום כדוגמת אחסון, מסדי נתונים, שירותי ספריה (LDAP), אבטחה (הזדהות, הרשאות, הצפנה, חתימה), שירותי העברת מסרים ותשתיות תקשורת.
- סדרת כללי עיצוב והמלצות מימוש (Best Practices) לאנשי הפיתוח אשר יאפשרו הבטחת שמירה על גמישות, חוסר תלות בפלטפורמה (.Net, Java), ויכולת פעולה בין פלטפורמות. במסגרת כללים אלו נכללים פרוטוקולי WS-* המתוקנים ע"י גורמי תקינה שונים, ונותנים מענה לניהול טרנזקציות, החלפת מסרים מאובטחת, ניהול מדיניות אבטחה, הזדהות ועוד.

נספח ג – מחשוב Grid - Grid Computing

מחשוב Grid מהווה טכנולוגיה עתידית (Emerging Technology) מבטיחה. יישומי Grid המנצלים את יתרונות התפישה הולכים ונבנים בשלב ראשון באקדמיה, אך יותר ויותר גם בארגונים עסקיים.

ישנם הגדרות רבות למושג של מחשוב Grid, אך ברובם הרעיון המרכזי טמון בעובדת היותו של הGrid "פדרציה" של משאבי מחשוב אשר תפקידה לזרז את ה Application Processing , תוך כדי וירטואליזציה (Virtualization) מלאה של משאבים אלו. בלב לבו המחשוב Gridי עוסק בעולם <u>המחשוב המבוזר</u> יחד עם אספקט חשוב של ניהול המשאבים.

חזון הGrid (Grid) הינו הפיכת המחשוב בתחום בעל סיבוכיות גבוהה אשר פיתוח פונקציה חדשה או שילוב פונקציות קיימות חייב הקמת מערכי תשתית גדולים או אינטגרציה ארוכה, לתחום המאופיין בדינאמיות בעלויות שוליות נמוכות לכל פונקצית קצה. בדומה לתשתית החשמל, שבה שימוש ברכיב חשמלי חדש שנקנה בחנות אינו מחייב אינטגרציה של חודשים: "כפי שאין לך עניין היכן מיוצר החשמל ומהיכן הוא מגיע, אלא אתה מעוניין באור בלבד, כך משתמשי המחשב לא מתעניינים היכן מתבצע עיבוד הנתונים, אלא רק בתוצר הסופי המופיע על מסך המחשב שלהם" [6].

על מנת להגיע לפריסה רחבה, ומוכנות של ארגונים להיכנס להשקעה בתחום ה–Grid נדרשת תמימות דעים על סט השירותים השונים הנכללים בליבת ה–Grid.

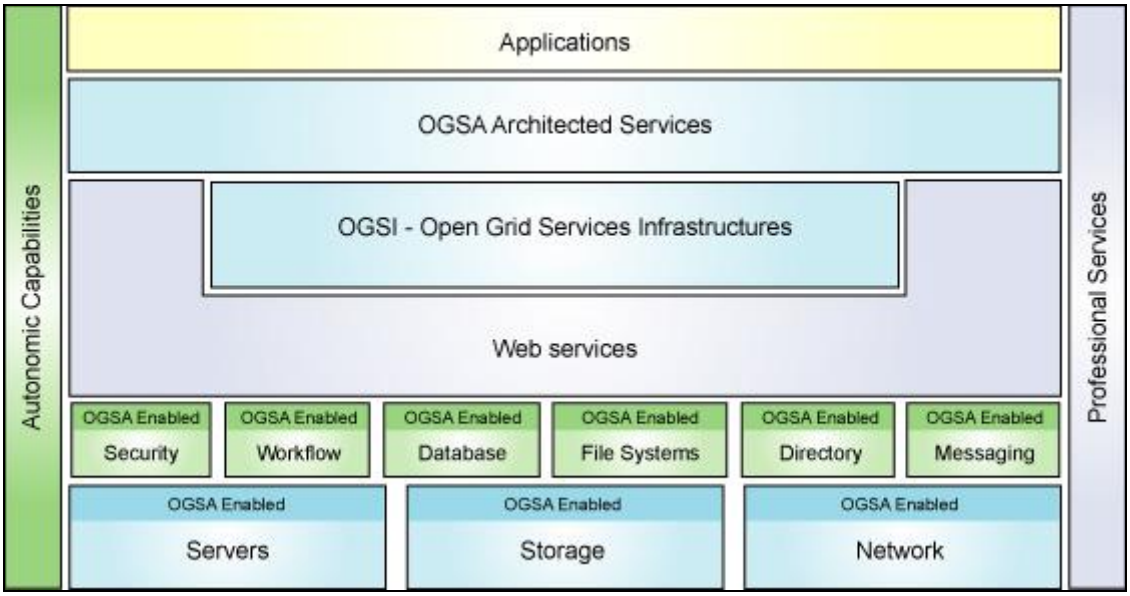
ה–Open Global Services Architecture (OGSA) הינה ארכיטקטורה שהוגדרה על-ידי קונסורציום בשם ה–Global Grid Forum, ומבוססת על הרחבה של Web Services[10,11]. שילוב של ה–OGSA עם פרוטוקולים מקובלים של Grid אפשרו את הגדרת ה– Open Grid Services Infrastructure (OGSI), אשר מהווה ארכיטקטורה אחידה לניהול ובנייה של Grid ומימוש של אפליקציות. הצורך בסטנדרטים פתוחים לצורך הגדרת האינטראקציות הללו הם המוטיבציה המרכזית ל OGSA.

ארכיטקטורת OGSA (Open Grid Service Architecture)

להלן תיאור קצר לגבי מטרות ארכיטקטורת ה-OGSA [15]:

- ניהול של משאבים בסביבות מבוזרות והטרוגניות.
- לספק איכות שירות (QoS) בצורה שקופה, שכן הטופולוגיה של הGrid הנה בדרך כלל מורכבת. האינטראקציה של משאבים בGrid מאופיינת בדינאמיות גבוהה, לכן חשוב לספק שירותי תשתית בזמינות ובשרידות גבוהה. בין אלו ניתן למנות שירותי הזדהות, בקרת גישה וכדומה[21].
- לספק תשתית אחידה לפתרונות של **Autonomic Management**. ה-Grid יכול להכיל מספר רב של משאבים עם מגון רחב של קונפיגורציות, אינטראקציות, ומצבים של כשלים ו/או שינוי בסטאטוס. במצב זה יכולות של מחשוב אוטונומי או רגולציה עצמית הנם צורך הכרחי.
- הגדרה של ממשקים פתוחים ופומביים - **OGSA** מהווה גוף סטנדרטים המנוהל על ידי ה **GGF standards body**. על מנת לתמוך באינטראופראביליות (Interoperability) בין מגוון רחב של משאבים, הGrid צריך להיות מבוסס על סטנדרטים ופרוטוקולים פתוחים.

- עשיית שימוש בטכנולוגיות ומוצרי אינטגרציה הקיימים בשוק. היוזמים של OGSA מקדמים שימוש בפתרונות קיימים בכל מקום בו הדבר אפשרי, אי לכך נכון להיום הבסיס ל OGSA הנו טכנולוגיית ה Web Services.
- ארכיטקטורת בת 4 שכבות:



איור 45 - ארכיטקטורת OGSA [15]

שכבות ה OGSA

קיימות 4 שכבות עיקריות בארכיטקטורת ה **OGSA**:

- **משאבים (Resources)** – משאבים פיזיים ומשאבים לוגיים.
- **שירותי רשת (Web Services) ו – OGSI Extensions** אשר מגדירות את שירותי הGrid.
- **שירותי** – **OGSA Architected Services**.
- **יישומי הGrid** (Grid Applications).

משאבים פיזיים ולוגיים

הקונספט של משאב הנו מרכזי בתפיסת המחשוב הGrid וה-OGSA. משאבים מכילים את היכולות של הGrid ואינם מוגבלים רק למעבדים בלבד. משאבים פיזיים כוללים: שרתים, אחסון, ורשתות. מעל המשאבים הפיזיים ישנם אוסף של משאבים לוגיים המספקים פונקציונאליות בסיסית על ידי וירטואליזציה ואיסוף (Aggregation) של משאבים בשכבה הפיזית לדוגמא: שרתי Middleware, System files, Workflow Managers וכדומה, הנם דוגמאות למשאבים לוגיים.

שכבת שירות הרשת (Web Services Layer)

שכבה זו הנה השכבה השנייה בארכיטקטורה, וכאן המקום לציין את אחד מקווי היסוד של ארכיטקטורת ה OGSA האומר שכלל המשאבים (פיזיים ולוגיים כאחד) בGrid ממודלים כשירותים אבסטרקטיים.

ה OGSI מגדיר ספציפיקציה מדויקת ל Grid Service תוך השענות על טכנולוגיות וסטנדרטים של Web Services. ה OGSI מנצל את מנגנונים כגון XML ו WSDL על מנת להגדיר ממשקים, התנהגויות ואינטראקציות סטנדרטיות בין אוסף של Grid Resources.

ה-OGSI מרחיב את ההגדרה של WS, על מנת לאפשר שירותי רשת מנוהלים ובעלי יכולת ניהול מצבים (STATE), תכונות הנדרשות עבור מידול משאבים בGrid.

שכבת OGSA Architected Grid Services

שכבת ה Web Services יחד עם הרחבות ה OGSI מספקת את התשתית הבסיסית לשכבה הבאה- שכבת ה Architected Grid Services.

ארגון ה GGF עובד כיום על הגדרת אוסף שירותי התשתית הארכיטקטוניים כגון : Program Execution, Data Services ו אוסף של Core Services נוספים. חלק מאלו כבר הוגדרו וחלקם האחר נמצא בתהליכי עבודה, מימושים מסחריים ראשוניים כבר נמצאים באקדמיה ואף בתעשייה. הופעתם של מימושים שונים באקדמיה לשירותי התשתית הללו גרמו להפיכתה של ארכיטקטורה OGSA לארכיטקטורת SOA שימושית עבור מגוון רחב של ארגונים ובעיות.

שכבת יישומי הGrid - Grid Applications Layer

במשך הזמן עם האבולוציה של שירותי התשתית הארכיטקטוניים (Grid Architected Services) אנו עתידים לראות את הופעתם של אוסף של יישומים חדשים אשר עושים שימוש באותם שירותי תשתית ארכיטקטוניים , יישומים אלו יהיו בנויים מ 4 השכבות שתוארו הלן.

נספח ד – יכולות ארכיטקטורת OGSA

נספח זה מתאר את סט היכולות המרכיבות את ארכיטקטורת OGSA, ואשר עונות לדרישות (Requirements) כפי שהוצגו בפרק 5 במסמך זה, בהם אשתמש לצורך מימוש תשתית ניהול ארגונית מבוזרת.

1. כללי

ארכיטקטורת OGSA מיועדת להקל על השימוש והניהול של משאבים בסביבה מבוזרת. בארכיטקטורת OGSA המושגים כגון, **"מבוזר" "הטרוגני" "ומשאב"** באים בשימוש במובן הרחב שלהם לדוגמא:

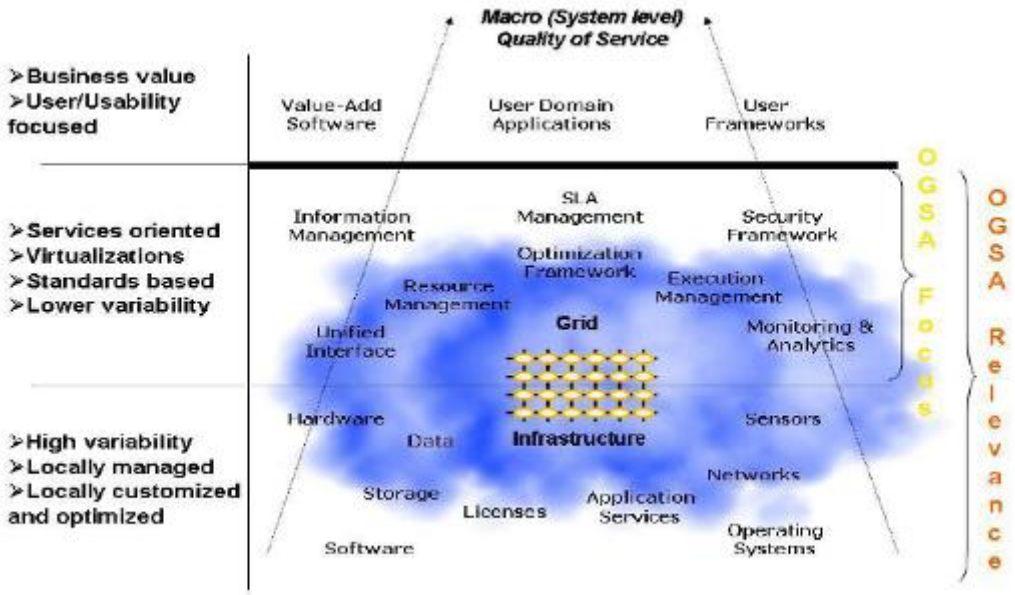
- המושג **"מבוזר"** יכול להתייחס מצד אחד לרצף גיאוגרפי של משאבים אשר מקושרים אחד לשני על ידי **"מארג קישוריות" (Grid-Fabric)** ומצד שני לרצף קישוריות גלובלית,המקשר מגוון רחב של משאבים על פני מספר מרחבים שונים.
- המושג **"משאבים"** מתייחס לכל רכיב, ישות או יחידת ידע אשר נדרשים לשם השלמת פעילויות בתוך או על המערכת. התועלת הנובעת מתשתית שכזו ממומשת על ידי סט של יכולות כפי שהן מתוארות באיור 46, איור זה מראה בצורה לוגית מופשטת ובמספר שכבות את הייצוג של חלק מהיכולות המרכיבות את תשתיות ארכיטקטורת OGSA.

השכבה הראשונה (תחתונה בציור) מתארת את המשאבים הבסיסיים (Base Resources). משאבים בסיסיים הם אותם משאבים אשר נתמכים על ידי הישויות אשר הונחו בבסיס המערכת או רכיבים אשר עלולים להיות פיזיים או לוגים, ויש להם רלוונטיות גם מחוץ לארכיטקטורה. דוגמאות לישויות כאלו כוללות: מעבדים, זיכרון, מערכות הפעלה, תהליכים.

הוירטואליזציה של משאבים אלו היא בצימוד הדוק לישויות אשר אותם הם מדמים, ולכן מרחב השמות בו נעשה שימוש בהקשר לישויות אלו משמש גם לייצוג ההדמיה שלהם.

משאבים אלו לרוב הם בבעלות וניהול מקומי אבל יכולים להיות בשיתוף מרוחק. התצורה וההתאמה האישית גם היא נעשית בצורה מקומית, וזאת בגלל שהישויות או הרכיבים יכולים להשתנות באופן שכיח ועל ידי מקורות שונים. משאבים אלו שונים מאוד במאפיינים שלהם, ברמת השירות שלהם, בגרסאות או בזמינות.

למרות שההבחנה במשאבי הבסיס נעשית בכדי לקשור את עקרונות הארכיטקטורה לרעיון של משאבים מסורתיים, חשובה להדגיש כי מבחינת הארכיטקטורה המתוארת במסמך זה אין הבחנה בין משאבי בסיס, תשתית, לשאר השירותים.



איור 46 - תרשים קונספטואלי של תשתית Grid [15]

השכבה השנייה (האמצעית) מייצגת רמה גבוהה יותר של וירטואליזציה והפשטה לוגית. וירטואליזציה וההפשטה מנחים לקראת הגדרה של מגוון רחב של יכולות אשר רלוונטיות לארכיטקטורת הGrid. יכולות אלו מספקות את התשתית הנדרשת בעבור תמיכה ברמה הגבוהה יותר של יישומים או משתמשים. סט היכולות הללו, כפי שמוגדרות בארכיטקטורה, הן כאמור יכולות סטנדרטיות ויחסית לא ניתנות לשינוי .

מנקודת מבט של משתמש הקצה, האופן שבו יכולות אלו ממומשות, ויתרה מכך מחוברות ומנוצלות על ידי יישומיי משתמשי המרחב, בעצם קובע את רמת השירות של המערכת ברמה הכוללת יותר של התשתית. חשוב להדגיש שהיכולות בדיאגראמה מייצגות רק דגימה של היכולות בארכיטקטורה, ותמונה רחבה יותר תוצג בהמשך פרק זה.

כאשר עוברים יותר בפירוט על יחסי הגומלין בין השכבה האמצעית לשכבה התחתונה כפי שמוצג באיור 46, ניתן לראות כי הארכיטקטורה מוכוונת השירותים, אשר מוטמעת בארכיטקטורת OGSA, מרמזת שמשאבים וריטואלים מיוצגים כשירותים ומשקיפים (peers) ביחס לשירותים אחרים בארכיטקטורה.

היחסים בין המשקיפים (peers) מרמזים על כך שהאינטראקציות יכולות להתבצע על ידי כל שירות בארכיטקטורה. יתרה מכך, שירותים בשכבה השנייה צריכים להשתמש ולנהל את וירטואליזציה (משאבים) בשכבה התחתונה בצורה כזו ששירות בודד יהיה מסוגל לספק יכולת זו. היחס ההדוק הזה מצביע על הבחנה דקה של השכבה התחתונה והאמצעית באיור 46. לפיכך ניתן להתייחס אל הישויות בשכבה התחתונה כחלק רלוונטי לארכיטקטורה.

השכבה השלישית (העליונה) הייצוג הלוגי מורכב מהאפליקציות ושאר הישויות שצורכות את היכולות של ארכיטקטורת הGrid כגון: משתמשים, פונקציות, תהליכים מרחביים ותהליכים עסקיים. אלו לרוב

שוכנים מחוץ לתיחום ארכיטקטורה הGrid, אבל הם גוזרים את ההגדרה מתוך תסריטי השימוש שהתשתית צריכה לתמוך בהם.
כל השכבות הללו אמורות לתקשר ולעבוד באופן טבעי ולספק את רמת השירות הנדרשת, ובגלל שמדובר ברמת השירות של המערכת כולה (כולל את שכבת היישומים אשר קובעת את חווית המשתמש) הדבר מעצב את רמת השירות ברמת מקרו כפי שרואים בחצים המנוקדים באיור 46.

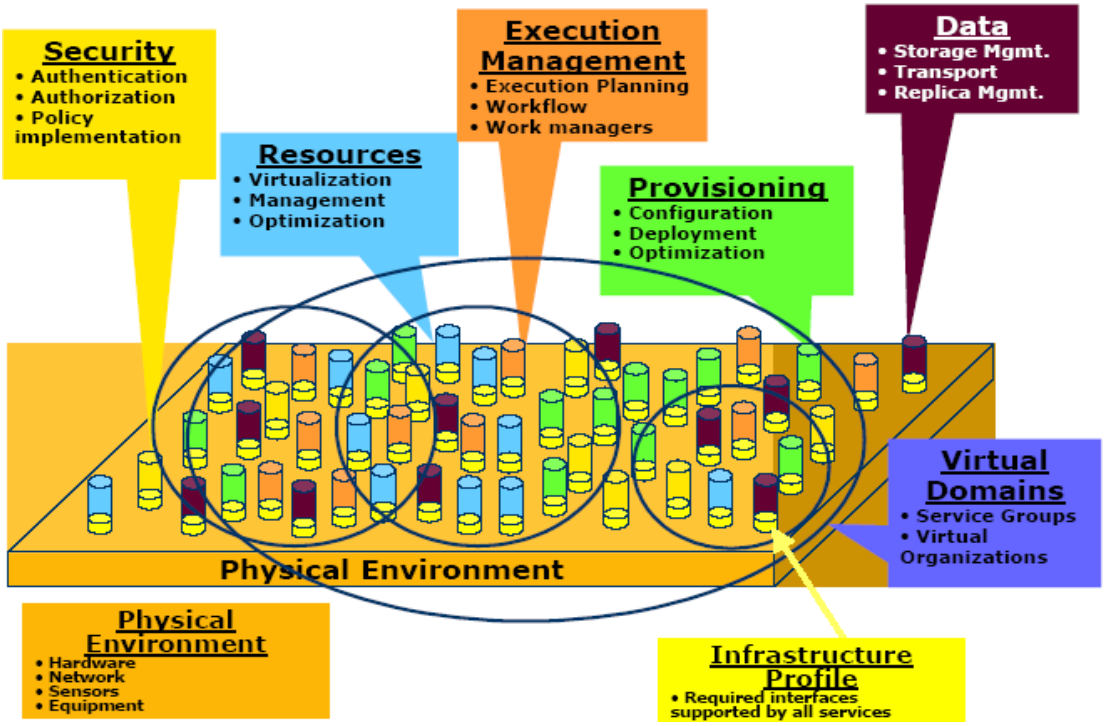
2. תיחום OGSA (OGSA Framework)

תיחום ליבת הארכיטקטורה הנו במימוש השכבה האמצעית. שירותים אלו חושפים, בצורה אינדיווידואלית וקולקטיבית את מצב המשאבים השייכים לשירותים אלו, ואת האינטראקציות בניהם, במסגרת הארכיטקטורה מונחית שירותים (SOA). שירותי התשתית של ארכיטקטורת OGSA מוצגים באיור 47 ו 48. בתרשימים אלו, הגלילים מיצגים שירותים בודדים. השירותים מבוססים על סטנדרטים של שירותי רשת, סמנטיקה, בנוסף להרחבה והעדכונים הרלוונטיים לGrid (Grid).
מספר נקודות נוספות לציון:

- אחת המוטיבציות החשובות ל-OGSA היא פרדיגמת הקומפוזיציה או גישת אבני הבניין, כאשר סט היכולות או הפונקציונאליות מאומצות כסט יכולות ראשוני מינימאלי, וזאת במטרה לספק את הצורך הפונקציונאלי הבסיסי. בעזרת פרדיגמת הקומפוזיציה נאפשר את הגמישות ויכולת התאמה קלה לשינויים כפי שנדרש מהארכיטקטורה.
- ארכיטקטורת OGSA מייצגת את השירותים, הממשקים, הסמנטיקה, ההתנהגות והאינטראקציות בין השירותים הללו. חשוב לציין שארכיטקטורות התוכנה הנפוצות בהם נעשה שימוש למימוש שירותים הנם מחוץ להגדרה והתחום אחריות של הארכיטקטורה.
- הארכיטקטורה לא ממפה את אותן שכבות הרלוונטיות למימוש של שירות בודד, אלה רק את שכבות השירות בעזרתן השירות מבצע את האינטראקציה עם השכבות עליהן הוא נסמך בצורה לוגית. למרות שיש דמיון לגישה מונחית עצמים, שירותים הם צופים (peers) בצימוד רופף, או לחלופין הם יחידה בודדת בתוך קבוצת אינטראקציות של שירותים, שמטרתן להגשים ולממש את היכולות של OGSA. כל זה נעשה דרך מימוש עקרונות הקומפוזיציה, או על ידי אינטראקציה עם שירותים (ראה איור 47 ו 48). לדוגמה: ניתן להגשים את יכולות התזמור (orchestration) של קבוצות של שירותים או שירותים בודדים כך שיתכן ששירותים יהיו מורכבים מקבוצה של שירותים המניעים את אותה קבוצה, כאשר שאר השירותים מספקים את הממשק והמנגנון אשר מאפשר להם להיות מנוהלים במסגרת אותו תזמור (orchestration). שירות ספציפי יכול להיות שותף במסגרת קבוצה של אינטראקציות, וזאת במטרה להשיג יכולות שונות ומגוונות. מצד שני, לא נדרש שכל השירותים ישתתפו ביכולת ספציפית בארכיטקטורה. שירותים יכולים להיות להשתתף בקבוצה וירטואלית

הנקראת virtual domain (ראה איור 47) (אם כשירות בודד או כקבוצת שירותים), או לחלוק נושא משותף או סביבת ניהול, כמו בארגון וירטואלי (Virtual Organization).

שירותי OGSA נדרשים להעריך את הסביבה הפיזית אשר יכולה לכלול רכיבים פיזיים ידועים מראש כגון חומרה ורשת ואולי אפילו ציוד פיזי כמו טלסקופ.



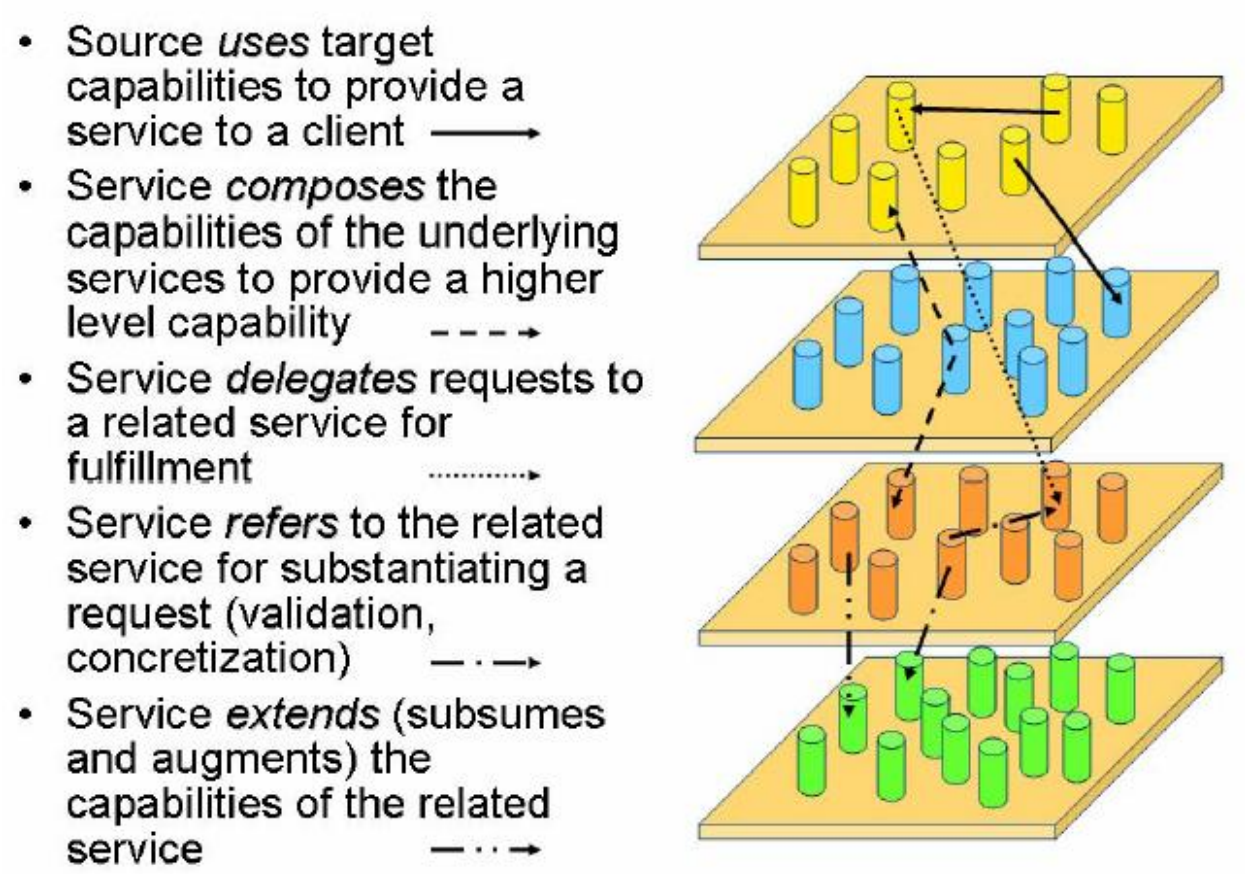
איור 47 - OGSA framework [15]

ניתן לצפות שיהיו סט של ממשקים, סטנדרטים וידע אשר שירותים חייבים לממש כדי להיות חלק מOGSA Grid. סט המימושים שמחוללים את ההתגשמות של OGSA נקראים שירותי תשתית או מארג (fabric) ה-Grid (Grid-Fabric). כפי שיצוין בסעיף הבא, נניח שהספציפיקציה WSRF, שכרגע נמצאת בפיתוח, תהיה חלק מהמארג Grid הראשוני. אתאר סטנדרטים נוספים אשר יכולים להיות חלק ממארג Grid בסעיף 3. איור 48 מראה תצוגה שונה מאיור 47 בהתמקדות על מספר קשרים אפשריים ואינטראקציות בין שירותי OGSA. הגלילים בציור מיצגים שכבת שירותים תשתיתיים פיזיים.

כל רמה (level) בציור היא אוסף המיצג יכולת בודדת, כאשר רמה אחת יכולה לממש את יכולת ניהול הביצוע ואחרת יכולה לממש את יכולת ניהול המידע. היחסים אשר מוצגים בין השירותים הספציפיים יכולים להשתרע על פני היכולות. מספר דוגמאות לסוגי היחסים אשר השירותים יכולים להציג הם שימוש, הרכבה, הורשה, ייפוי כוח, הפניה וכולי. אספקטים של אינטראקציות שירותים למטרות נוספות כגון ניהול והצהרה על פרופיל השירות יכולים להיות ממודלים בעזרת היחסים של השירותים.

3. שירותי תשתית (Infrastructure Services)

אחת המטרות המרכזיות בהגדרה של OGSA הנה להגדיר סט רכיבים עקביים ומקושרים אשר ביחד מתייחסים לדרישות שזוהו בפרק 5, וזאת כחלק מההקשר של ארכיטקטורה מוכוונת שירותים (SOA). חובה עלינו לבצע הנחות יסוד לגבי שירותי התשתית בארכיטקטורה, עליה נבנות הצהרות מוצקות והנחות יסוד לגבי שירותים ברמות הגבוהות יותר.



איור 48 - מיפוי היחסים בין השירותים [15]

קיימות הרבה דוגמאות לספציפיקציות אשר ניסו להיות כלליות מידי ולא השיגו את מטרתן. לדוגמה, פרוטוקול ה-CORBA נכשלה להשיג קישוריות בגלל ששמות השירותים היו תלויים במימוש. ולכן, רק כפי שהתכנון של TCP, DNS, ועוד פרוטוקולים מושפעים מהמאפיינים של התשתית IP עליהם הם מושתתים, התכנון שלנו ל-OGSA מושפע מהמנגנונים השוכנים בארכיטקטורה. ההנחה הראשונית היא שהעבודה על OGSA תורמת לפיתוח של סדרת ספציפיקציות טכניות הקשורות לעולם ארכיטקטורת שירותי הרשת [WS-Architecture] וזאת מכיוון שארכיטקטורת OGSA מסתמכת על היכולות הקיימות של ארכיטקטורת שירותי הרשת, אך במקרים רבים משלימה את ארכיטקטורת שירותי הרשת ומרחיבה אותה, כך ששירותי הרשת יהיו חלק מהGrid.

ניתן לראות את OGSA כפרופיל לאפליקציות מבוססות על סטנדרטים של שירותי הרשת (WS). בחרתי בכך שכן ארכיטקטורת שירותי הרשת היא הארכיטקטורה המתאימה ביותר לאמץ בצורה נרחבת סטנדרטים מסחריים אשר מאפשרים את הפונקציונאליות הנדרשת למערכות Grid.

בחירה זו בשירותי הרשת כתשתית משמעותה שאני מניח שמערכות בתקן OGSA ויישומים נבנים בהתאם לעקרונות ארכיטקטורה מונחית שירותים, ושל הממשקים אשר מוגדרים ב WSDL.

לרגע נניח ש WSDL-1.1 והמעבר העתידי ל WSDL-2.0 אשר מתוכנן בשלב מאוחר יותר כבר מוכנים. נניח ש XML הוא הטכנולוגיה העיקרית (למרות שיש להכיר בכך שיידרש אולי ייצוג אחר במקרים שנדרשים ביצועים גבוהים), ונניח ש SOAP הוא פורמט החלפת המסרים העיקרי עבור שירותי OGSA. בנוסף, נהיה מעוניינים לפתח הגדרות שירותים אשר תואמים את הקישוריות ואת הפרופיל על-פי WS-I.

בזמן שאלו עובדים בתוך סביבת שירותי הרשת, זה ברור שהסטנדרטים של שירותי הרשת כפי שהם מוגדרים כרגע אינם ולא מתוכננים לענות על דרישות הGrid. במקרים מסוימים, ספציפיקציות קימות עלולות להידרש בהרחבה או עדכון. ולכן ארכיטקטורת OGSA מעורבת בהגדרה של WSDL 2.0 ובמעבר על אבטחת שירותי הרשת וכן בספציפיקציות רלוונטיות, וכן אזהה במסמך זה תחומים נוספים אשר מרחיבים את הספציפיקציות כנדרש.

במקרים אחרים, דרישות הGrid מעודדות הגדרות של שירותים חדשים לחלוטין. תחום מפתח בדרישות הGrid אשר מעודד הגדרה של ספציפיקציה חדשה הוא תחום האבטחה. נושאי האבטחה עולים ברמות שונות של OGSA.

אאמץ שימוש בסטנדרטי האבטחה של WS-Security כדי לאפשר לשירותי OGSA לבצע ולבקש בצורה מאובטחת שירותים כגון authentication, authorization, message protection מקצה לקצה.

הגנה מקצה לקצה נדרשת בתסריטים השונים של הארכיטקטורה, ולאלו OGSA חייבת לספק מנגנוני הגנה ברמה גבוהה כגון XML, הצפנה, וחתימה דיגיטאלית וזאת בנוסף (או במקום) לאבטחה בין נקודה לנקודה כגון TLS ו IPSec. בנוסף לאבטחה ברמת המסר, תשתית קישוריות שרכיבי האבטחה עצמם יהיו מיוצגים כשירותים לדוגמה, שירות הרשאות יכול להשתמש בתקן WS-Agreement ביחד עם התקן המפותח על ידי OASIS ובשימוש ב SAML ו XACML לייצג אבטחה והרשאות גישה היכן שמתאים, OGSA תאמץ, או תגדיר שירותי אבטחה אלו.

תחום מפתח שבו דרישות הGrid עודדו ספציפיקציות חדשות, ספציפיקציות בעלות רלוונטיות מעבר לתסריטי הGrid, הוא בתחום ייצוג המצב הנוכחי (state), ומניפולציה על state. ובאופן ברור, אניח כאבני בניין את הממשקים וההתנהגויות אשר מוגדרים על ידי WSRF. כאשר WSRF מגדיר גישה לעיצוב, נגישות, וניהול המצב; במטרה לקבץ שירותים; ולבטא שגיאות. למנגנונים אלו יש את כל התפקידים הבסיסיים לבנות מערכות Grid.

בנוסף, WSRF בא בשימוש או בא בחשבון במגוון של מודולי ניהול משאבים מערכות ומאמצי גיבוש סטנדרטים על ידי OASIS WSDM ולכן הסטנדרטים הקשורים ב-OGSA אשר מבוססים על WSRF ממוקמים היטב ומקושרים היטב עם המאמצים של גופי הסטנדרטים. ולבסוף, אבסס את יכולות ההתרעות ואירועים על הסטנדרט של WS-Notification כך שניתן יהיה לקבל ולהירשם על אירועים רצופים לגבי שינויים במצב הרכיבים.

4. ניהול הפעלת השירותים (Execution Management Services)

יכולות ניהול הפעלת השירותים (EMS-OGSA) מתייחסים לבעיות של אתחול וניהול יחידות עבודה. לדוגמה יחידת עבודה יכולה לכלול אפליקציות OGSA או אפליקציות מורשת (Legacy) שאינן OGSA כגון (Database Servlet, Application Server) וכיוצא בזה.

4.1. מטרות (Objectives)

הדוגמה הבאה ממחישה מקצת מהנושאים אשר מטופלים על ידי שירותי ה - EMS.

לדוגמה אקח תסריט של יישום הזקוק לשירותי Cache. שאלות רבות נשאלות, האם היא צריכה להשתמש בשירות קיים או ליצור אחד חדש? במידה ונוצר שירות חדש, היכן למקם אותו? וכיצד לקנפג אותו? כיצד המשאבים יהלמו את צרכיו? (CPU, Disk space, Memory), איזה סוג מדיניות שירות ה-Cache יספק? ולאיזה סוגי מדיניות הוא זקוק?

תסריט אחר הנו מצב בו משתמש רוצה להפעיל אפליקציות מורשת (legacy). במצב זה בתחומים מגוונים כגון ביו-אינפורטיקה, אלקטרוניקה, תעופה וחלל, ובין אם בשירותים חשבונאים, ישנם סוג של אפליקציות אשר מקבלות מידע כקלט ,לרוב בקבצים ובפרמטרים ומיצרים פלט.

רוב האפליקציות הללו מייצרות מופעים אשר רצים כפי שנקרא "בחישוב מקבילי". דוגמה טובה לכך נתן למצוא בעולם הביו-אינפורמטיקה ונקראת BLAST, אפליקציה זו משווה רצף פרוטאין או DNA מול בסיס נתונים ומייצרת רשימת תוצאות זהות. דוגמה אחרת מעולם התעופה הנו יישום Overflow – המבצע חישוב דינאמיקת הנוזל, קוד אשר משתמשים בו לסימולציות מטוסים.

בכל מקרה הסוגיות אליהן יש להתייחס כאשר מפעילים אפליקציות מורשת (Legacy) כוללים:

- היכן האפליקציה תרוץ ?
- כיצד קבצי המידע וקבצי ההרצה יוצבו במקומם?
- מה יקרה אם ההפעלה תיכשל?
- האם לאתחל את ההפעלה, ואם כן איך?

בצורה פורמאלית יותר שירותי "EMS" מתייחסים להפעלה של יחידות העבודה, זה כולל את הצבתם, אספקתם, וניהול תקופת החיים שלהם. בעיות אלו כוללות אך אינן מוגבלות ל:

- ***Finding Execution Candidate Locations*** - באלו מיקומים נמצאות יחידות עבודה שיכולות לספק שירות בהתחשב במגבלות המשאבים כגון CPU והגבלות מדיניות.
- ***Selecting Execution Location*** - ברגע שברור היכן נמצאות יחידות העבודה שיכולות לפעול, השאלה היא היכן הן יכולות לפעול? מענה לשאלה זו כולל אלגוריתמי בחירות שונים אשר מבצעים אופטימיזציה בדרך להשגת מטרות שונות ואכיפת מדיניות שונה או רמת שירות שונה.
- ***Preparing For Execution*** - רק מכיוון שיחידת עבודה יכולה לעבוד במקום מסוים לא אומר שבהכרח היא אמורה לפעול שם ללא התקנה כל שהיא. התקנה יכולה לכלול

קונפיגורציה של קבצים בינאריים וספריות, הצבה של מידע, ופעולות שונות נוספות אשר נדרשות להכנת הסביבה לריצה.

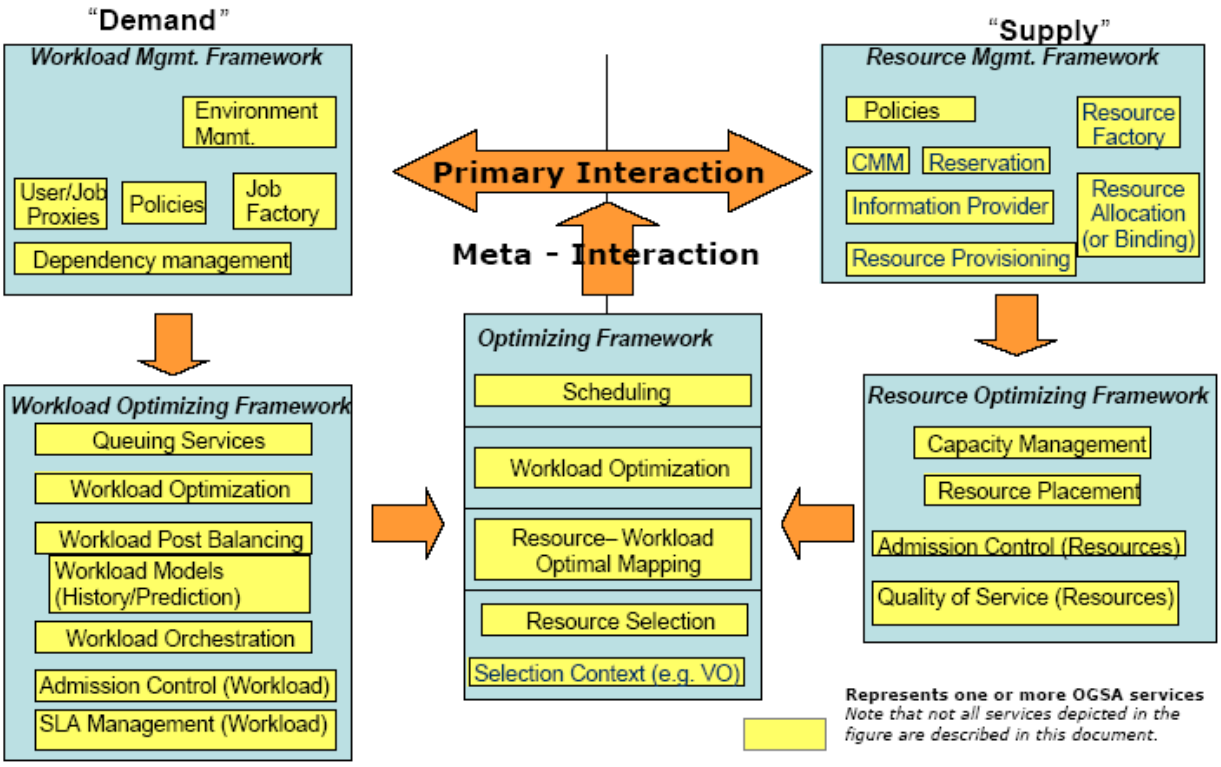
- ***Initiating the Execution*** - כאשר הכול מוכן, ביצוע ההפעלה למעשה ופעולות אשר כרוכות בכך, כגון רישום בכל המקומות המתאימים.
- ***Managing the Execution*** - לאחר שההפעלה החלה, חובה לנהל ולנטר אותה עד לסיומה. ולתת מענה לשאלות: מה עושים בכישלון? אי עמידה בהסכמים. האם להתחיל את ההפעלה במקום אחר? מה קורה עם תמונת המצב? האם צריך לשמור את תמונת המצב בכל מיני נקודות ציון? האם ההפעלה היא חלק מסוג מסוים של סכמת איתור תקלות ושיקום מתקלות?

אלו הם הנושאים העיקריים אשר מטופלים במסגרת שירותי ה EMS. ניתן לראות שבכך כוסה מכלול המשימות, ומחולל אינטראקציות עם הרבה שירותי OGSA כגון Provisioning, Logging, Registries and Security אשר יוגדרו כיכולות אחרות ב OGSA.

חשיבותו של שירות ה-EMS נובעת מכך שאין אנו יכולים להניח שהסביבה היא סטאטית ואשר עושה שימוש ב - Registries כמו UDDI. אנו מצפים שמערכות Grid יהיו בשימוש בצורות רבות ושונות כאשר מגוון המשאבים, ועומסי העבודה אשר מופעלים על המשאבים הם משתנים בצורה רבה ומגוונת. לדוגמה בסביבת מחשוב דינאמית שניתן לחזות בה, סט המשאבים שבשימוש על ידי אפליקציה מסוימת יכולים להשתנות במשך הזמן, במטרה לספק את דרישות האפליקציה ורמת השירות שעלולה להגדיר צריכת משאבים מרוחקים באופן זמני. באופן דומה להגיב לתקלות בלתי צפויות, ועדיין לעמוד ברמת שירות שהובטחה, עלול להצריך מציאת משאבים והפעלתם מחדש. היכולת הנפוצה היא היכולת לנטר ולהגיב בצורה דינאמית לדרישות אלו עד לסיום.

4.2. גישה (Approach)

הפתרון לבעיית ה EMS כולל סט שירותים אשר ביחד מקבלים ביטוי כסדרה של רכיבים אשר ניתנים להחלפה. תסריטי שימוש שונים יכולים להשתמש בחלקים שונים של שירותים אלו במטרה להשיג את המטרות שלהם. באופן כללי ניתן להתייחס אל שירותי EMS כספקים וצרכנים של שירותים.



איור 49 - EMS מטה דטה מחולק לעולם המספק ולעולם הדורש [15]

שירותי ה-EMS מאפשרים לאפליקציות לבצע גישה מנוהלת למשאבים, ללא קשר למיקומם הפיזי או המנגנון גישה שלהם. שירותי EMS הם המפתח להקל את הגישה למשאבים בעבור משתמש הקצה, על ידי התאמה אוטומאטית של הדרישות האפליקטיביות למשאבים הפנויים. שירותי EMS כוללים מספר שירותים אשר עובדים יחד. בהמשך מתוארים שירותים אלו. אך לפני שאמשיך, מספר הבנות והערות.

- דבר ראשון לא כל השירותים יהיו בשימוש באותו זמן. מספר מימושי הGrid אינם זקוקים לחלק מהשירותים, או שיכולים להחביא חלק מהשירותים בתוך שירותים אחרים, ולא להפוך אותם לזמינים באופן ישיר. באופן כללי, ניסינו לפרק את היכולת הזו לשירותים מאחורי פונקציות אלו שזיהנו שוב ושוב במימושי Grid שונים, במילים אחרות פונקציות אשר מספר מימושים שונים צריכים, צורכים, ומתממשקים אליהם.
- שנית, כרגע מדובר בהגדרה ראשונית והמטרה היא לא להגדיר את כל השירותים. המטרה היא לזהות את רכיבי מפתח ואת האינטראקציות שלהם.
- שלישית, נדגיש שהגדרות אלו של שירותים יהיו ברי יישום כשירותי רשת, ולא רק הפעלות של משימות (Jobs).
- ולבסוף, אניח את הקיום של "resource handle" מונח אבסטרקטי למשאב והמצב (State) המיוחס לו, אם יש. אניח גם שקיים מנגנון שמקשר בין משאב לכתובת שלו. כתובת של משאב מכילה פרוטוקול ספציפי אשר בעזרתו ניתן להתקשר עם המשאב. אשתמש במונח RH בהתייחסות ל-resource handle. כתובת משאב מכילה מידע

שהוא ספציפי לפרוטוקול שדרכו מתקשרים עם המשאב, ואני אשתמש בקיצור RA בהתייחס ל resource address.

4.3. שירותי EMS (EMS Services)

שלושת הסוגים העיקריים של שירותי ה EMS הם:

- משאבים אשר ממדלים עיבוד, אחסון, קבצי הפעלה, ניהול משאבים, ותחזית.
- ניהול משימות (Jobs) ושירותי ניהול.
- בחירת משאבים אשר מביאות להחלטה היכן להפעיל יחידות עבודה. נניח את הזמינות של שירותי ניהול המידע, שירותי אבטחת מידע וכן שירותי מעקב. על האינטראקציות עם שירותים אלו ארחיב בשלב מאוחר יותר במסמך זה.

4.4. משאבים (Resources)

4.4.1.מיכל שירות (Service Container)

מיכל השירות [container] מכיל ישויות פעילות, בין אם הם Jobs (יוסבר מאוחר יותר) או שירותי רשת פעילים. מיכל יכול לדוגמה להיות שירותי תורים, UNIX host, סביבת J2EE או אוסף של מכלי שירות [Containers]. ל-Containers יש מאפייני משאב אשר מתאר מידע סטאטי כגון איזה קבצי הפעלה הם יכולים להשתמש, גרסאות מערכת הפעלה, ספריות, מדיניות, וסביבת אבטחת מידע, וכן גם מידע דינאמי כגון הבטחת איכות. מיכל השירות מיישם חלק מממשקי הניהול של ניהול משאב עלפי תקן WSDM וכן מצופה הרחבת הממשקים אשר מספקים שירותים נוספים מעבר למיכל השירותים הבסיסי. מכלי השירות ה-Containers יהיו בעלי מגוון יחסים למשאבים נוספים אשר חשופים לצרכנים. לדוגמה למיכל יש את קשר עם מיכל מידע אשר מצביע שישויות פעילות בתוך מיכל מסוגלות לגשת למידע בתוך מיכל מסוים. משאבים מנוהלים נוספים יכולים להיות מותקנים על גבי מערכות הפעלה או רשת פיזית. לבסוף, אנו מצפים שמיכלים ישתמשו בשירותי הזמנת משאבים מראש, שירותי מעקב, שירותי מידע,שירותי ניהול משימות ועבודה, ושירותי תחזית ואספקה.

4.4.2.שירותי ניהול שמירת מצב (Persistent State Handle Service (PSHS))

שירות זה מבצע מעקב אחר מיקום המצב [state] . שירות כזה יכול להיות ממומש באופנים שונים כגון מערכת קבצים, בסיס נתונים, מערכת אחסון נתונים היראכית. לשירות PSHS יש פונקציות כגון Get Resource Handle. סוג ה RH הנו תלוי בדרך בה הוא שמור הלכה למעשה. כתובת המשאב Resource Address יכולה להיות נתיב במערכת הקבצים או מפתח של ערך בבסיס הנתונים. הדבר החשוב הוא שניתן להשתמש ב RA ישירות כדי לגשת למידע. שירות PSHS מממש את ממשקי הניהול של WSDM managed resource.

ניתן לצפות גם להרחבת הממשקים אשר מספקים שירותים מעבר ל Container הבסיסי .
שירות ה-PSHS יהיה גם בעל פונקציות לניהול המשאבים שבתוכם, כולל העברה בין PSHS למשנהו. זה ישמש להעברה ולאפליקציה. עוד דרך לחשוב על PSHS הוא כמאגר Metadata אשר מספק מידע לגבי כיצד מידע יהיה בשימוש בצורה יעליה לדוגמה מפתח בבסיס נתונים או מסלול לקובץ.

4.5. ניהול משימות (Job Management)

4.5.1.משימה (Job)

שירותי ה EMS (Execution Management Services) של OGSA מגדירים את המושג משימה (Job) אשר מגיע בירושה מהמונח ההיסטורי בעולם המחשוב של Job.
הישות Job מכילה בתוכה את כל מה שיש לדעת על יחידת עבודה (לדוגמה מופע של אפליקציה או שירות). ה-Job הוא היחידה המנוהלת הקטנה ביותר. הוא מיצג את אספקט הניהול של יחידת עבודה. הוא אינו דומה לאפליקציה עצמה שרצה, או לאספקט הריצה של יחידת עבודה. משימה (Job) מממש חלק ממשקי הניהול של WSDM. השם של מופע ה - Job ניתן על ידי RH ייחודי. והוא נוצר ברגע שהוא נדרש, למרות שלא הוקצו משאבים באותו רגע. ה Job מבצע מעקב אחר שלבי הביצוע (התחיל, עוכב, ואתחל, הופסק, סיים). הרבה מהמידע נשמר ב Job document. ה-Job Document מתאר את מצב המשימה- לדוגמה, תיאור ההפעלה (JSDL), ההסכם אשר הושג, סטאטוס הג׳וב, מידע לגבי המשתמש ומספר הפעמים ש - Job הופעל.
אין אנו כוללים ״במצב״ מידע בנוגע לאפליקציה הספציפית כגון מצב הזיכרון הפנימי של תכנית בזמן ריצה. ה-Job Document חושף את עצמו כמאפייני משאב של הג׳וב.

4.5.2.מנהל משימות (Job Manager)

מנהל המשימות הנו שירות ברמה גבוהה יותר בהיררכיה, השירות מכיל את כל האספקטים של הפעלת המשימה, או קבוצת המשימות מההתחלה ועד סיומם. קבוצת משימות יכולה להיות מורכבת מסוגי ג׳ובים שונים.
ה-Job Manager (JM) יכול להיות פורטל אשר מבצע אינטראקציה עם משתמשים ומנהל את המשימות בשמם.
ה-JOB MANAGER לרוב יבצע קישוריות עם שירותי Execution Planning, שירותי התקנה וקונפיגורציה, שירותי ניטור ובקרה. יתרה מכך הוא צריך להתמודד עם תקלות ואתחול מחדש ויתכן שיידרש לתזמן משימות למשאבים, תפקידו לאסוף הסכמים והזמנת משאבים מראש קודם להפעלה.
ה-JOB MANAGER יממש את ממשקי WSDM Collection שהם אלו ממשקים לניהול ישויות מנוהלות. ה-WSDM Collection יכול לחשוף את הפונקציות שלו ואלו יכולות להיחשף על-ידי החברים בקבוצה שלו.

ה-JOB MANAGER אחראי לתזמר את השירותים אשר בעזרתם אפשר להתחיל משימה או קבוצת משימות, לדוגמה משא ומתן על הסכמים, קישוריות עם מכלים, ובקרה ושירות מעקב. הוא יכול לשקף את המאפיינים של משאב הג׳וב מתוך אותם משימות שהוא מנהל לדוגמה: תור אשר מקבל משימות מבצע עליהם תעדוף, מפיץ אותם למשאבים שונים למחשוב.

ה-JOB MANAGER עוקב אחרי משימות, ויאתחל מכלים (Containers) של משימות אשר מתאימים לביצוע משימה.

- רכיב פורטל אשר מבצע אינטראקציה עם משתמשי הקצה לאיסוף מידע ודרישות על משימות , וכמו כן מתזמן את המשימות ומחזיר תוצאות למשתמש הקצה.
- מנהל זרימת עבודה (Workflow Manager) אשר מקבל סדרה של תיאורים, דרישות QoS, יחסי הגומלין בניהם ומידע התחלתי תפקידו לנהל ותזמן את זרימת העבודה עד סיום תוך כדי התגברות על כשלים לא צפויים
- מערך של משימות אשר לוקחים משימה זהה עם הבדל קטן בפרמטרים ומנהל אותם עד סיום.

4.6. שירותי סלקציה (Selection Services)

4.6.1.שירותי תכנון ההפעלה (Execution Planning Services)

שירות תכנון ההפעלה (EPS) הוא שירות אשר בונה מיפויים אשר נקראים מתזמנים (Schedulers), מיפויים אלו מתזמנים בין משימות (Jobs) ומשאבים (Resources). מתזמן הוא יחס של מיפוי בין שירות ומשאב, יתכן והוא עם מגבלת זמן.

מתזמן יכול להיות מורחב עם סידרה של אופציות שאומרות בערך כך: ״אם המתזמן הזה נכשל תנסה אחד אחר במקום״. שירות ה-EPS בדרך כלל ינסה לבצע אופטימיזציה לנושאים כגון זמן ההפעלה, עלויות, אמינות, ועוד.

שירות ה-EPS אינו מגלם דמות של מתזמן, הוא פשוט מחולל אותו. השיפור של מתזמן נעשה על ידי ה-JOB MANAGER. שירות ה-EPS לרוב ישתמש בשירותי מידע ו Candidate Set Generator (ראה למטה). לדוגמה קודם יקרא ל-CSG לקבל רשימת משאבים, ואז יקבל מידע יותר עדכני ורלוונטי על אותה רשימת משאבים מתוך שירותי המידע, ואז יפעיל אופטימיזציה ליצר את המתזמן.

4.6.2.שירותי איתור משאבים זמינים (Candidate Set Generator)

הרעיון הבסיסי הוא פשוט יחסית, המטרה לזהות ולאתר סדרת משאבים עליהם יחידת עבודה יכולה לרוץ,הכוונה היכן ניתן להריץ את יחידת העבודה? ולא איפה ובאיזה מיקום היא תופעל?

נושא זה כולל ויכול להתייחס לנושאים כגון קבצי הרצה זמינים, דרישות אפליקציה מיוחדות (לדוגמה GB4 זיכרון ו40 GB דיסק זמני, xyz ספריות מותקנות), הגבלי אבטחה (איני מעוניין לתת למשימה שלי לרוץ על משאב אם הוא אינו בדוק ומאושר). ה-CSG מיצר סדרה של מכלים (Containers) אשר בהם ניתן להריץ ג׳וב אשר קיבל את שמו מה-RH. אנו מצפים ש-CSG יופעלו על ידי EPS או שירותים

אחרים כגון JOB MANAGER אשר מבצעים פונקציות EPS. אנו מצפים מה-CSG להשתמש בשירותי מידע לבצע גישה למשאבים ולאתר את החלקים המתאימים של Job Document, ולבצע אינטראקציה שירותי Provisioning ומכלי שירותים ולהחליט האם ניתן לקנפג מיכל לביצוע מסוים.

4.6.3.שירותי הזמנה מראש (Reservation services)

שירותי הזמנה מראש מנהלים את הזמנת המשאבים, מתקשרים עם שירותי גביה (יתכן וצריך לגבות כסף בעבור השימוש בשירות). הוא אינו יכול להיות שירות נפרד, אלה ממשק לקבל ולנהל הזמנות מראש מתוך מכלי השירות (Containers) ומשאבים נוספים. ההזמנות עצמן הן חוזה מוסכם. שירות הזמנות מציג ממשק אחיד לכל הסוגים של המשאבים על הGrid. משאבים שניתן להשתמש בהם יכולים לכלול משאבי מחשוב כגון CPU, זיכרון, גרפיקה, מקום אחסון, רוחב פס, ומכשירים כגון רדיו או טלסקופ.
הזמנה יכולה להיות איחוד או קבוצה של הזמנות ברמה נמוכה, אשר יבואו במשא ומתן או ימכרו על ידי מתווך צד שלישי. שירותי הזמנה מראש באופן כללי באים בשימוש על ידי מגוון שירותים: JOB MANAGER יכול לחולל הזמנות לקבוצת משימות אשר מנוהלות על ידי EPS יכול להשתמש בהזמנות כדי להבטיח תכנית הפעלה בעבור משימה ספציפית. יכול להיות מקרה בו יצירת הזמנה תהיה קשורה בתכנון ותחזית בעבור משימה.

4.7. אינטראקציות עם שאר רכיבי OGSA (Interactions with the rest of OGSA)

חלק זה מפרט את האינטראקציות של EMS עם שאר החלקים של OGSA

4.7.1.התקנה וקינפוג של שירותים (Deployment & Configuration Service)

לרוב, לפני ששירות או מיכל מידע (Data Container) יכול לבוא בשימוש על ידי יחידת עבודה, חובה לקנפג או לקשרו לעוד משאבים. לדוגמה לפני שמריצים BLAST על גבי שרת, המשתמש חייב להבטיח שקובצי ההרצה של BLAST והקונפיגורציה שלו נגישים לשרת. דוגמה נוספת היא קונפיגורציה והתקנה של אפליקציה מורכבת, בסיס נתונים מתאים, או התקנה של LINUX על מכונה כצעד ראשון למחשוב של משאב.

4.7.2.קונבנציית שמות (Naming)

ה-EMS-OGSA משתמש ב Naming-OGSA, לדוגמה במנגנון תורים בעבור משימות אשר יש נקודות בדיקה ויכולת לאתחול מחדש בעבור זמינות או איזון עומסים, הכתובת של המשימה עלולה לציין את מיקום המשימה במכונה מסוימת, השם האבסטרקטי מזהה את המשימה ללא קשר למיקומו ובצורה אוניברסאלית, לדוגמה, שם אבסטרקטי צריך להיות זהה לפני ואחרי שהמשימה נדדה. השם האנושי יכול להיות שם בעל משמעות למשתמש וקצר אשר וניתן לזהותו ביחס להקשר שהוא בא בשימוש.

4.7.3.שירותי מידע (Information Service)

בקצרה, שירותי מידע [Information Services] הם בסיס נתונים של מאפייני מידע על שקשורים למשאבים. כחלק מ-EMS, שירותי מידע באים בשימוש על ידי שירותים רבים ושונים: לדוגמה, מכלים (Containers) צריכים להפיץ מידע לגבי המאפיינים שלהם כדי שה-CSG יהיה מסוגל להעריך את התאימות של המכל לביצוע המשימה, שירות EPS צריך לקרוא מידע אודות מדיניות לגבי VO מתוך שירות מידע שירות ה-PSHS עצמו יכול להיות ממומש בעזרת שירותי המידע. לא מוגדר כיצד שירות המידע מקבל את המידע שלו, למרות שאנו מצפים שעדכניות המידע תהיה מאפיין על המידע.

4.7.4.ניטור (Monitoring)

לפעמים פשוט להפעיל משהו זה לא מספיק. אפליקציות לרוב צריכות להיות מנוטרות בעיקר מסיבות של התאוששות מתקלות ומסיבות של הבטחת איכות. לדוגמה: מצב בו התנאי שבמקור בחר במתזמן [Scheduler] השתנה, ויתכן שזה מצביע שיש לבחור בתזמון חדש.

4.7.5.איתור שגיאות ושירותי התאוששות (Fault-Detection and Recovery Services)

שירותי איתור תקלות והתאוששות יכולים להיות או לא להיות חלק משירותי הניטור, שירותים אלו יכולים לכלול תמיכה לניהול סכמה פשוטה בעבור פונקציות Stateless אשר מאפשרות דו-שיח בנושא ביצועים ושימוש במשאבים, וגם סכמות קצת יותר מורכבות אשר מנהלות נקודות בדיקה והתאוששות של Jobs בעלי Thread בודד,וכן גם ניהול של אפליקציות מורכבות אשר המצב [state] שלהם מבוזר.

4.7.6.שירותי ביקורת תיעוד וחיוב (Auditing, billing and logging services)

שירותי ביקורת תיעוד וחיוב הם קריטיים להצלחה של OGSA. וכוללים את היכולת למתזמנים לבצע אינטראקציה עם משאבים במטרה לזהות את המחירים שלהם, וכן למשאבים לתקשר עם שירותי החיוב והחשבונאות שלהם.

- **מדידה** [Metering] משתמש ביומן להמשיך ולבצע מעכב על השימוש במשאבים.
- **ביקורת** [Auditing] משמש את היומן בצורה עקבית [Persistent].
- **חיוב** [Billing] שוב עוד שירות שלא מוגדר על ידי OGSA, אשר משמש את שירותי הביקורת ואו המדידה ליצר חשבונות, או חיובים.

4.7.7.חשבונאות (Accounting)

כמו כרטיס אשראי, חלק מהמשאבים צריכים לראות אם למשתמש יש את היכולת לשלם. המתזמן [scheduler] עלול להידרש לבצע אינטראקציה עם שירותי החשבונאות, וכן משאבים מסוימים כגון מיכלים [containers].

4.8. תסריטים אפשריים

הדרך הכי טובה להבין את השירותים היא לראות איך הם באים בשימוש במטרה להשיג דוגמאות מוצקות. בחרנו שלוש דוגמאות:

- System Patch Tool
- Deploying a Data Caching service
- Legacy application execution
- איסוף תמונת מצב ממשקים

4.8.1. תסריט 1: System Patch Tool

לרוב עדכוני מערכות הפעלה או ספריות עדכון צריכות להיות מופצות על גבי מספר רב של שרתים. ניתן להשיג מטרה זו על ידי מספר דרכים. אחת הדרכים הנפוצות היא להריץ תוכנת מחשב המבצעת פקודה אחר פקודה ללא קומפילציה [script] על כל שרת מארח אשר אליה הועתקו הקבצים הרלוונטיים. תוכנות מחשב אלו [scripts] לרוב מופעלות על ידי שימוש ב rsh או ssh ונקראות על ידי מעטפת מערכת ההפעלה וזה רץ על גבי מספר שרתים ומבצע את העדכונים. אלטרנטיבה נוספת, שרתים יכולים בצורה מתוזמנת לבדוק אם הם צריכים עדכון, ואם כן להריץ תוכנת מחשב שתבצע את העדכון.

כאשר משתמשים בשירותי EMS פעולה זו יכול להתבצע במספר דרכים. בהנחה שגרסת מערכת ההפעלה היא חלק ממטה-מידע אשר מאוכסן במאפייני המכל [container] וזה נאסף על ידי שירותי המידע, המטרה היא לעדכן את כל מערכות ההפעלה אשר לא מעודכנות. אולי הדרך הפשוטה לטפל בבעיה היא דבר ראשון לבצע שאילתה משירותי המידע ולקבל את רשימת המכלים [Containers] אשר גרסת מערכת ההפעלה שלהם היא מתחת לסף מסוים ואז לבצע אינטראקציה עם JOB MANAGER ולהריץ שירות עדכון על כל מיכל אשר ברשימה. במקרה הזה JOB MANAGER אינו צריך לבצע אינטראקציה עם שירותי תכנון ההפעלה (EPS) וזאת מכיוון שהוא יודע היכן הוא רוצה להריץ את השירות. במקום, ה JOB MANAGER מבצע גישה ישירה עם כל מיכל-ומבצע התקנה וקינפוג לשירות וזאת במטרה להפעיל את שירות העדכון על המיכל (יתכן ששירותי ההתקנה והקונפיגורציה ידרשו להתקין קודם את שירותי העדכון)

4.8.2. תסריט 2: A Data Cache Service

דמיינו שירות מטמון מידע [Cache Service] אשר שומר מידע (קבצים, קבצי הפעלה, בסיס נתונים, וכו') בעבור מספר צרכנים, ומשמר את הקשר עם ההעתק המקורי. כאשר הצרכן מבקש שירות מטמון, הוא מסוגל להעביר מצביע למטמון קיים או ליצור מטמון חדש וזאת בהתאם למיקום, עומסי עבודה, ועוד. כאשר ההחלטה בוצעה לאתחל מופע חדש,שירות ה-EPS מופעל במטרה להחליט היכן לשמור את מידע המטמון. שירות EPS משתמש ב CSG על מנת להחליט היכן אפשר להריץ את השירות ביחס למבקש השירות.

4.8.3.תסריט 3 : A Legacy Application

הדוגמה השלישית ממחישה תסריט נפוץ. נניח שמשתמש רוצה להריץ אפליקציית מורשת כלשהי, יתרה מכך נניח שהמשתמש מבצע אינטראקציה עם פורטל או מנהל תורים. ישנם ארבעה צעדים בסיסיים להתחיל את משימת הרצת האפליקציה.

1. הגדרת המשימה. אילו קצבי הזנה? מה רמת השירות? מתי הפעולה אמורה להסתיים? ואיזה חשבון לחייב?
2. זיהוי ובחירת המשאבים הזמינים אשר נדרשים לביצוע ה JOB.
3. הפעל מתזמן וכל מה שנדרש לדוגמה חיזוי משאבים חיוב משאבים.
4. ניטור על כל אורך החיים של הג׳וב בתלות על רמת השירות של הפעולה והפעלות חוזרות במידה ויש כישלונות מאיזה שהיא סיבה.

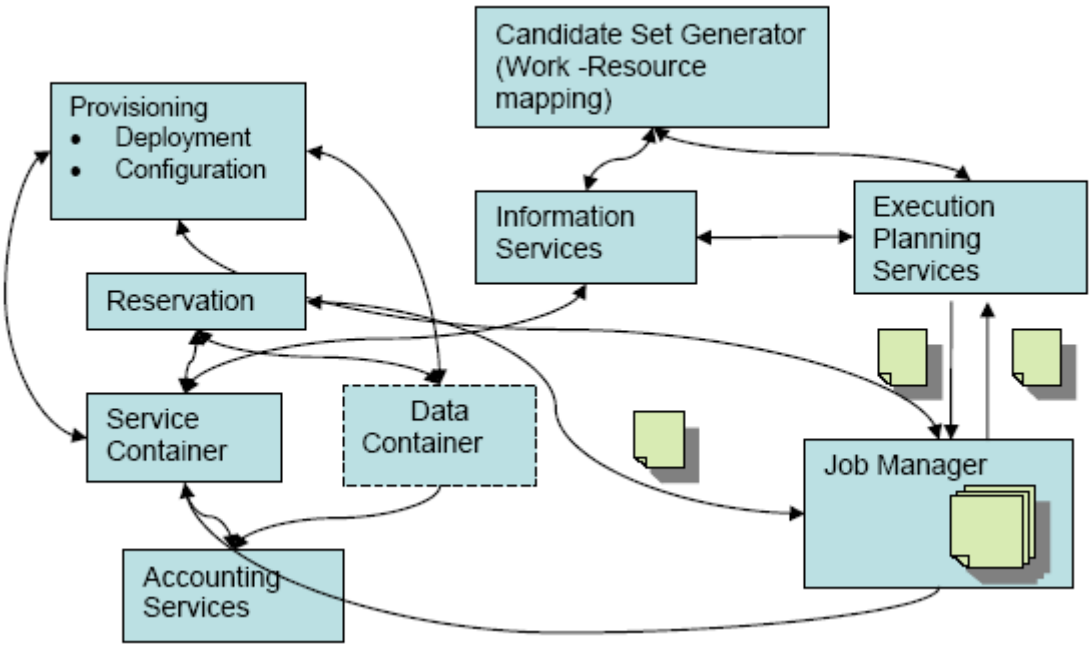
כדי להגשים את התסריט, ה-Job Manager מייצר שירות מורשת [legacy] חדש עם תיאור משימה מתאים. ה-Job Manager קורא לשירות ה EPS בכדי לקבל תזמון לביצוע. ה EPS בתורו קורא ל CSG, אשר קורא לשירותי המידע וקובע היכן הג׳וב ירוץ על-פי מאפייני מדיניות וזמינות. ה-EPS בוחר מיכל השירות, אחרי שקודם בדק שמיכל השירות מכיל מידע נכון, ה-EPS מחזיר תזמון לפעולה ל Job-Manager. ה Job Manager מבצע אינטראקציה עם שירותי הזמנת משאבים מראש ושירותי ההתקנה וקונפיגורציה במטרה להקים את סביבת ההרצה למשימה. מיכל השירות מאתחל את הג׳וב. שירותי התיעוד משמשים לחשבונאות וביקורת ומעקב. כאשר הג׳וב מסתיים ה- Job Manager מיודע על כך על ידי מיכל השירות. אם הג׳וב מסתיים בצורה לא תקינה כל התהליך מופעל שוב (ראה איור 50).

4.8.4.איסוף תמונת מצב ממשקים

תסריט איסוף תמונת מצב ממשקים, מהווה חלק בלתי נפרד מהצרכים הנגזרים בעולם הקישוריות הבין-זרועית, במערך זה קיימים צרכנים שונים (כגון: לוג, בסיס נתונים, מערכת קצה, מנהל מערכת) המעוניינים לאסוף בזמן אמת את תמונת מצב (state) של הממשקים (המשאבים) השונים.

מאפייני המשאבים יכולים להיות לדוגמה (שם זרוע, מערכת קצה, מציאות, סוגי ישויות, פרוטוקול טכני, מסרים שעברו או לא עברו בתקופה מסוימת).

שירותי ה-EMS ב OGSA יכולים להוות בסיס מצוין למימוש התסריט , כאשר משתמשים בשירותי EMS פעולה זו יכולה להתבצע במספר דרכים בהנחה שמאפיינים אלו הם חלק ממאפייני המשאבים הפזורים ברשת ומידע זה נאסף על ידי שירותי המידע, המטרה היא לאתר ולנהל את אותם ממשקים (משאבים) שהמצב שלהם מתחת לסף מסוים ואז לבצע אינטראקציה עם ה JOB MANAGER ולהריץ את ה Job המתאים לעדכון המצב בעבור אותם ממשקים, הג׳וב מבצע אינטראקציה ישירה עם כל ממשק (משאב) ומבצע פעולות בדיקה ניהול וקינפוג של (הממשק) המשאב ובכך ה Job מבצע אינטראקציה ישירה ואינטימית עם המשאב ומקבל את תמונת המצב של כלל המשאבים הרלוונטיים בהתאם לחתך המאפיינים.



איור 50 - שימוש בשירותי EMS להפעלת Legacy Job [15]

5. שירותי גישה לנתונים (Data Services)

5.1. מטרות (objectives)

הצורך בשירותי גישה לנתונים הנו שינוע לכל מיקום שבו הוא נדרש, ניהול העתקים, הרצת שאילתות ועדכונים, והמרת מידע לפורמט חדש.

שירותי גישה לנתונים גם מספקים יכולות ניהול מטה-מידע (Metadata) במטרה לתאר את שירותי המידע של OGSA או שירותים אחרים, ובעיקר את מקור המידע עצמו.

לשם הדגמה נניח ששירות ניהול ההפעלה (EMS) זקוק לגישה למידע אשר שמור במקום אחר, במקרה זה מתעוררות שאלות רבות כגון:

האם הוא משתמש בשירותי גישה לנתונים (Data Services) על מנת לגשת אל המידע אשר נמצא במקום מרוחק או לחילופין מעתיק את המידע למכונה שעליה הוא רץ? האם הוא שומר מידע מטמון במכונה מקומית? האם המידע זמין במספר מקומות? מהן הגבלות המדיניות או הבטחת השירות ששירות המידע מספק?

או בתסריט אחר נניח כי שירות איחוד מידע מעוניין להגדיר סכמה של נתונים אשר שמורה במספר מקומות שונים. במקרה זה נשאלת השאלה איך שאילתות על גבי הסכמה הזו וממופות לתשתית המשאבים? היכן להפעיל קישור או המרה של נתונים? היכן המידע צריך להיות מסופק? ומהי רמת השירות הנדרשת? למעט מספר מחלקות של מטא-מידע, היכולות אינן מגדירות משמעות למידע מסוים. שאר השירותים ב-OGSA לדוגמה שירותי מידע (Information Services) אשר משתמשים בשירותי גישה לנתונים (Data Services) יכולים להוסיף את המשמעות שלהם.

5.2. מודלים (Models)

5.2.1.סוגים של משאבי נתונים (Types of Data Resource)

משאב נתונים הוא ישות אשר יכולה להתנהג כמשאב או מאגר של מידע. האופי שההטרוגני של הGrid מגדיר שהרבה מהסוגים השונים של המידע צריכים להיתמך. זה כולל אך לא מוגבל לקבצים שטוחים או קבצי xml. הצורה הפשוטה ביותר של מידע היא קובץ עם תצורה ספציפית, כמו רשומות באורך קבוע. ניתן לגשת לקבצים אלו בקריאה וכתיבה קונבנציונאליים חלק מהפורמטים תומכים בשאילתות דומות לשאילתות בסיסי נתונים. דוגמאות כוללת ערכים מופרדים בפסיקים וניתן לבצע שאילתות כמו בטבלאות רלציוניות, וקבצי XML בעזרת XQUERY. הגישה לנתונים יכולה להיות כהרחבה ותמיכה לפורמטים נוספים כמו:

- **STREAMS** - האפשרות של רצף מידע אין סופי נקרא stream שירות הגישה למידע כולל שאילתות והמרות על גבי Streams.
- **DBMS** - מספר סוגים של ניהול בסיס נתונים כחלק מהGrid. וזה כולל בסיס נתונים רלציוניים, XML, ו OO.
- **קטלוג** - קטלוג ומעקב אחרי שירותי מידע אחרים. דרך פשוטה של קטלוג היא ספריה, אשר מכילה רשימה של קבצים. ספריות מכוננות דומות למבנה הירארכי.
- **גזירה** - חלק מהנתונים הם תוצאה של שאילתות א-סינכרוניות או המרות של מידע אחר. הגזירות האלו לרוב מנוהלות בדומה ל-streams.
- **שירותי המידע -** עצמם יכולים להיות משאבי מידע בעבור שירותים אחרים, כמו שהתקן, חיישן או תוכניות מייצרים מידע.

5.2.2.תסריטי דוגמה (Example Scenarios)

יכולות שירותי הגישה לנתונים הם בסיסיים לGrid. בחלק זה מתוארות מספר דוגמאות וכיצד הן יכולות לבוא בשימוש ולתמוך במגוון הפעילויות.

- **Remote Access -** הדרך הפשוטה ביותר להשתמש בשירותי הגישה של OGSA היא לגשת למידע מרוחק על גבי הGrid. השירותים מסתירים מנגנוני תקשורת מצד הלקוח; ואם נדרש הם יכולים להסתיר את מיקומם המדויק של מידע המרוחק.דרך לאופטימיזציה של מנגנון כזה היא על ידי יצירת מטמון (CHACE) של חלק מהמידע על גבי המשאב המקומי.
- **Staging -** כאשר מפעילים משימות על משאב מרוחק, שירותי המידע בדרך כלל באים בשימוש במטרה להזיז מידע קלט אל המשאב אשר מוכן אל ה Job, ולאחר מכן להזיז את התוצאה אל המיקום המתאים.
- **Replication – מאפשרת** לשפר את זמינות ולהפחית את זמן הגישה, אותו מידע יכול להיות מאוחסן במספר מקומות על גבי הGrid.

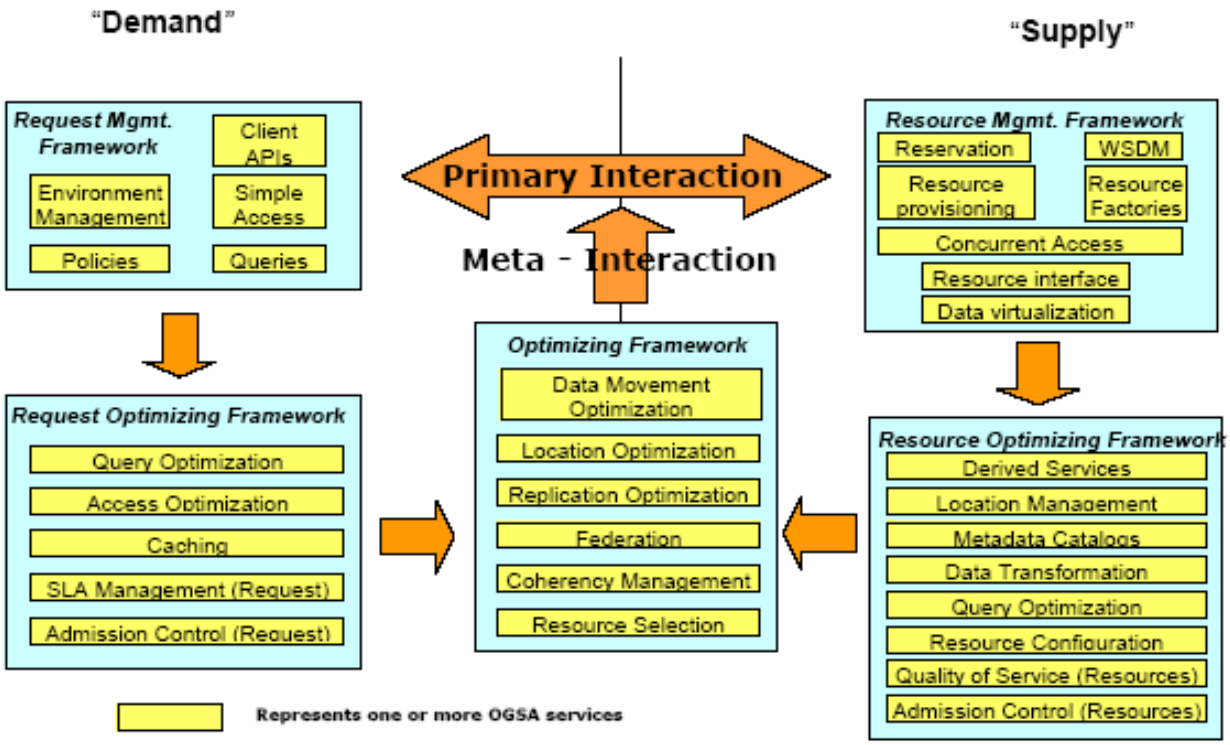
- **Federation -** שירותי המידע של OGSA מאפשרים את היצירה של מידע וירטואלי אשר מקשר מידע ממספר מקורות מידע אשר נוצר ומתוחזק בצורה נפרדת. כאשר הלקוח מבצע שאילתה למציאת המשאבים הוירטואליים, השאילתה מהודרת לתתי שאילתות ופעולות (operations) אשר מחצינות את המידע המתאים, ובצורה כזו מתבצע איחוד של תתי השאילתות לכלל שאילתא אחת המתבצעת בצורה מקבילית ומחזירה תוצאה מאוחדת יחידה.
- **Derivation** - שירותי המידע של OGSA תומכים בחילול אוטומטי ממשאב מידע אחד לשני.
- **Metadata -** מידע אשר מתאר את שירותי המידע של OGSA או מידע אחר הוא בסיסי לתפקוד הGrid. בצורה הפשוטה, זה עוד שימוש לשירותי המידע של OGSA.

בנוסף, OGSA מספק תמיכה לתחזוקת הקשר בין המידע עצמו ובין מידע העל שלו

5.2.3.צריכה ואספקה של מודל העל (Supply & Demand Meta Model)

כחלק משאר החלקים של OGSA, ניתן להתייחס אל ארכיטקטורת שירותי המידע כמורכבת מספק וצרכן. ספקים מספקים את משאבי המידע כגון קבצים, בסיסי נתונים או שירותים. הם מספקים את וירטואליזציה הספציפית למשאב כפי שמתוארת למטה.

בצד הספק ישנם גם שירותי המרת מידע ומטה-מידע, ושירותים אשר אוכפים את המדיניות והסכמי השירות. בצד הדורש ישנם שירותים אשר מייצרים בקשות וגישה אופטימאלית, מנהלים את רמת השירות. וכן המקשרים בין שני הצדדים הם השירותים אשר מעתיקים, מאחדים, ומבצעים אופטימיזציה למיקום המידע. ניתן לראות כל זאת באיור 51.



איור 51 - ניהול המידע והמטה דטה של עולם הצרכן ועולם הספק [15]

5.3. יכולות פונקציונאליות (Functional Capabilities)

פרק זה מתאר יכולות פונקציונאליות אשר מסופקות על ידי שירותי המידע של OGSA חלקים שונים של השירותים הללו אמורים לממש יכולות שונות.

5.3.1.שקיפות ווירטואליזציה (Transparency and Virtualization)

מערכת מבוזרת בנויה מאוסף מגוון של משאבים. משאבים אלו משתמשים בדרך כלל במודלים שונים לייצור של המידע, מדיה פיזית שונה לשמירת המידע, אמצעים שונים לניהול המידע, סכימה שונה לתיאור המידע, פרוטוקולים וממשקים שונים שמאפשרים גישה למידע, המידע יכול להישמר מקומית או מרוחק, המידע יכול להיות חד-ערכי או העתק , הגיש למידע יכולה להיות באופן קבוע או על-פי דרישה.

שירותי המידע של OGSA. יכולים להגדיר וירטואליזציה של המשאבים. וירטואליזציה היא ראייה אבסטרקטית אשר מחביאה את ההבחנות הללו ומאפשרות למשאב המידע להיות מותאם ללא קשר אליהן.

למרות ששירותי מידע מאפשרים ללקוחות להתעלם מהם, חלק מצרכנים זקוקים ויכולים לנצל את ההבחנות האלה לשם ביצוע שאילתה או להגדיר מיקום של מידע ספציפי ולהשתמש בו.

לקוח אחד יכול להשתמש בגישה פרימיטיבית למשאב, כאשר אחר יכול לכונן את פרמטרי הביצועים של משאב המידע. בכדי לתמוך בלקוחות כאלו, שירותי המידע של OGSA מאפשרים לעקוף את ממשקי הוירטואליזציה ולגשת למשאב ספציפי באופן ישיר. השכבות הללו מאפשרות ללקוחות לבחור קומבינציה של כוח ופישוט שמתאימה להם. לדוגמה, גישה בסיסית של כתיבת I\O מספקת ממשק

אשר מאפשר כתיבה וקריאה. השירותים מאפשרים פעולות המתוחכמות ואופטימאליות כמו מטמון, העתקה, והעברה אופטימאלית של מידע. אותם צרכנים שזקוקים לשליטה יותר מפורטת יכולים להשתמש בממשקים הישירים של משאב ספציפי לבצע את הפעולות האלה וזאת בתשלום השקיפות כתמורה.

5.3.2.נגישות צרכן בעזרת API (Client APIs)

משתמשים רבים היו מעוניינים להשתמש בשירותי המידע של OGSA עם אפליקציות מורשת (Legacy) אשר משתמשות ב API כמו NFS, CIFS, JDBC, ODBC, ADO, POSIX IO or XQuery. על-פי רוב זה יהיה יותר מידי יקר או לא פרקטי לשכתב את הלקוחות להשתמש בממשקים חדשים.
במקרים אלו שירותי OGSA יכולים בקלות לבצע הדמיה של אותם API למרות ש OGSA עצמה לא תגדיר כאלו פתרונות.
בדרך כלל מעטפות מסוג זה, ימפו את הפעולות של אותו API קיים בהתאם למסר אשר נדרש לשלוח לשירות המידע של OGSA. ולכן הם עלולים לספק גישה רק לחלק מהפונקציונאליות הכוללת של OGSA. ההרחבה של הפונקציונאליות של OGSA אשר זמינה תלויה בתיחום של אותו API רלוונטי.

5.3.3.סוגי מידע הניתנים להרחבה ופעולות (Extensible data type support and operation)

מכיוון שלא ניתן לחזות מראש את כל משאבי המידע שיבואו בשימוש על ידי שירותי המידע של OGSA. השכבות בארכיטקטורה המתוארות למעלה (איור 51) מאפשרות את ההוספה של משאבים חדשים בעתיד. באופן דומה, לא ניתן לחזות את הפעולות שנדרש לבצע על משאב נתון.
השכבות בממשקים המתוארים למעלה (איור 51) מספקות שירותים אשר מאפשרים פעולות נוספות מעבר לאלו שצוינו ב OGSA. שכבת הוירטואליזציה נותנת ללקוחות את האופציה להתעלם מהרחבות אלו, כאשר צרכנים אלו דורשים לעקוף את הוירטואליזציה כנדרש.

5.3.4.ניהול מיקום המידע (Data Location Management)

שירותי המידע של OGSA מציעים העברת מידע אמינה ממקום אחד לשני,על ידי יצירת העתק של המידע המקורי או העתקתו למיקום חדש בצורה מושלמת. מידע יכול להיות טמון (Cached) בכל מקום נדרש וזאת במטרה להימנע מהעברות מיותרות נוספות.
מידע מטמון (Cached Data) יכול להיות מקונפג במושגים של ניהול זמן חיות, עדכון, עקביות. וכן מידע יכול להיות משוכפל במספר העתקים, וכן הגדלת הזמינות שלו דרך יצירת עודפים. שירותים בעבור המשתמשים האינדיבידואלים מאפשרים להם להעלות ולנהל את המידע שלהם בעזרת ההתקנים שצוינו למעלה. הגדרות האבטחה שולטות בגישה המותרת למשתמשים נוספים.

5.3.5.גישה פשוטה (Simple Access)

גישה פשוטה לשירותים מספקת פעולות של כתיבה וקריאה (לוגיים) של רצף בתים המשכיים ממקור מידע כלשהו. המשאב יכול להיות מקומי או מרוחק: ממשק הוירטואליזציה מסתיר את הפרטים של מקום המידע. וזה מאפשר לשירות לבצע אופטימיזציה גם של מקום המידע וגם בגישה למידע.

5.3.6.שאילתות (גישה מובנית) (Queries Structured Access)

שירותי הגישה למידע של OGSA מספקים מנגנון לביצוע שאילתות למשאבי מידע מובנים. במקרה הפשוט אלו יכולים להריץ שאילתת SQL על גבי בסיס נתונים רלציוני, שאילתת XML על גבי בסיס נתונים XML , או regular expression על גבי קובץ טקסט.

שירותים אחרים יכולים ליישם חקירת טקסט על גבי מספר מסמכים, או שאילות מבוזרות על גבי בסיס נתונים מאוחד. שאילתות סינכרוניות מחזירות מידע כתשובה לבקשה, כאשר שאילתות א-סינכרוניות חושפות את המידע הנגזר כמשאב חדש. שירותים יכולים להחזיר תוצאות של שאילתה לסדרה של שירותים אחרים. שירותי שאילתות יכולים לבצע אופטימיזציה של השאילתה לפני שהיא נשלחת אל המשאב. המשאבים יכולים לבצע אופטימיזציה נוספת של השאילתה ויכולים להתמודד עם נושאים של גישה מקבילית למידע.

שירותי איחוד מידע מנתחים כל שאילתה אשר מתקבלת ומייצרים תת שאילתה להרצה בתצורת משאב מבוזר. הוא יכול להחליט היכן התווך של העיבוד מתבצע במטרה להפחית את תעבורת הרשת. איחוד המידע חייב לספק מידע רלוונטי לשינויים והרחבות במנועי זרימת עבודה ולאפשר להם לתזמן פעולות בצורה יעילה.

5.3.7.המרות (Transformation)

שירותי מידע עצמם יכולים להמיר מידע. לדוגמה, הם יכולים להמיר מידע מפורמט אחד לשני, או לסנן אותו לפני שמעבירים אותו או מעדכנים אותו. הם יכולים לתמוך בשאילתות שמורות (stored-procedures) אשר מופעלות בתוך השירות, ובכך מתפקד השירות כסוג של מיכל Container. ההמרות יכולות להיות מופעלות ישירות על ידי פעולה מסוימת או שהן יכולות להיות מתוכנתות להפעלה אוטומטית כאשר תנאי מסוים מתקיים.

5.3.8.עדכון מידע (Data Update)

שירותי המידע של OGSA מספקים קשת של מנגנונים לעדכון מידע, בהסתמך על הסמנטיקה של משאבי המידע. בעבור הקטלוג, הפעולות כוללות יצירה שינוי שם ומחיקה. לקבצים מובנים ובסיסי נתונים הפעולות כוללות עדכון של כניסות בבסיס הנתונים. לרצף מידע ושאר הפעולות של קבצים הפעולות מוגבלות להוספת מידע חדש. שירותי מידע יכולים להגדיר התנהגויות טרנזקטיביות בעבור פעולות עדכון.

למשאב מידע יכולה להיות גרסה משוכפלת או ריבוי מקורות לגזירת מידע, העדכון של גרסאות השירות יכול לכלול את הגרסה המשוכפלת. במקרה כזה ובמקרה שבו קיימים מספר צרכנים המעדכנים את אותו מקור מידע, השירותים יכולים לממש מגוון סוגים ושיטות של תחזוקה, ובלבד שהצרכן תמיד יקבל את התוצאה של השאילתה אותה הוא ביצע.

5.3.9.מיפוי למנגנוני האבטחה (Security Mapping Extensions)

מערכות ניהול בסיס נתונים לרוב מממשות מנגנון אבטחה מתוחכם. חלק מהן מספקות מגוון רחב של פעולות אפשריות ושליטה בגישה ברמה של טופולוגיה אינדיבידואלית. ולכן שירותי המידע של OGSA תומכים בהרחבה של תשתית האבטחה הסטנדרטית של OGSA.

5.3.10. קונפיגורציה של משאבי מידע (Data Resource Configuration)

משאבי מידע לרוב מספקים אופציות קונפיגורציה מתוחכמים. אלו יכולים להיות זמינים לצרכנים דרך שירותי המידע של OGSA בנוסף השירותים יכולים לספק פעולות קונפיגורציה של הוירטואליזציה אשר מספקת פעולות נוספות של קונפיגורציות המשאבים. לדוגמה, שירות מידע רלציוני יכול לאפשר לטבלה מסוימת מבסיס הנתונים אשר שוכנת בתשתית להיות חלק משירותי המידע, או ל view על גבי בסיס הנתונים להיות זמין כטבלה בתוך אותה וירטואליזציה.

5.3.11. מטא-מידע (Metadata)

שירותי מטא-מידע הם שירותי מידע אשר מאחסנים מידע לגבי שירותי המידע של OGSA או לגבי מידע אחר. ה-OGSA מספקת תמיכה לתחזוקת הקשרים בין שירותי OGSA למידע העל אשר מתאר אותם. תמיכה זו כוללת מידע על מבנה המידע, כולל קישורים לסכמות אשר מתארות את המידע. לחלק מהשירותים זה לא פרקטי, כאשר המידע יכול לכלול סכמות אשר משתנות בתכיפות גבוהה, במקרים אלו סכמות מידע יהיו מסופקות על ידי השירותים עצמם.

5.3.12. מקור (Provenance)

מטא-מידע (Metadata) לגבי השירותים יכול לכלול מידע לגבי מקור ואיכות המידע. נושא זה יכול להתקיים בכל רמה של המשאב או ברכיבים אשר מרכיבים אותו, לפעמים ברמה של אלמנט בודד. וזה בתורו מצריך מהשירות או תהליכים אחרים אשר מיצרים את המידע לנהל תחזוקה של אמינות מידע-העל.

מידע איכותי על מקור המידע יכול לאפשר לבנות נתונים מחדש על ידי מעקב אחרי זרימת המידע אשר יצרה אותם במקור.

5.4. מאפיינים (Properties)

מאפיינים הם יכולות לא פונקציונאליות. מאפיינים מהווים אספקט בארכיטקטורה אשר תקף למגוון רחב של שירותים. היכן שיכולות הפונקציונאליות מוגדרות על ידי הכניסות בממשקי השירותים, למאפיינים לא פונקציונאלית יש תפקיד סמנטי גלובלי יותר.

5.4.1.גידול (Scalability)

שירותי המידע של OGSA מאופיינים ביכולת גידול , הללו בנויים לטפל במערכי מידע גדולים, מספר רב של מערכי מידע, זרימת מידע גדולה, כולל ריבוי אתרים.

5.4.2.הבטחת איכות השירות (Quality of Service)

שירותי המידע של OGSA יכולים לממש מגוון רמות של QoS, כגון העברה אמינה.

5.4.3.עקביות (Coherency)

כאשר מידע מועתק, נגזר או מוטמן (Cached), קיים מגוון של פעולות זמינות פעולות שליטה ואסטרטגיות משקיפים נתמכים להתקנה של מידע על וקטלוגים, ומציאת קונפליקטים בעדכונים.

5.4.4.ביצועים (Performance)

שירותי מידע של OGSA מעוצבים ומתוכננים להקטין למינימום את התנועה וההעתקה של המידע. זהו פקטור חשוב בביצועים הכוללים של הGrid. שירותי העברת מידע משתמשים במידע ניתור כגון רוחב פס, דפוסי שימוש וגדלים. דבר זה מאפשר להם לבחור בגישה להעברת מידע תוך כדי תמיכה והבטחת רמת השירות הנדרשת.

5.4.5.זמינות (Availability)

שירותי המידע של OGSA מספקים יכולות לייצוג אירועים ברשת או שגיאות. לדוגמה, שירותי שאילתה אשר יכולים להיות מקונפגים להחזיר חלק מתוצאה כאשר חלק מהמשאב זמין.

5.4.6.הגבלות משפטיות ואתיות (Legal and Ethical Restrictions)

שירותי המידע של OGSA יכולים לתפקד בסביבה אשר כוללת מגוון של מדיניות משפטית ואתית אשר משפיעות על הפעולות שלהם. לדוגמה חלק מהמדיניות יכולות להגביל ישויות אשר יכולות לגשת למידע אישי ולהגביל את הפעולות אשר ניתן לבצע מפאת חיסיון.
נושאי פרטיות עלולים להגביל שאילתות אשר מבוצעות על ידי יחידים, למרות שבחלק מהמקרים המדיניות היא לאפשר שאילתות אשר מחזירות תוצאות לגבי קבוצות ככלל, לדוגמה ממוצע ההכנסה או השכר הכולל. מידע לרוב מוגן על ידי הגבלות זכויות יוצרים ,דבר אשר מגביל את הזכות לייצר עותקים של המידע. מנגנוני האבטחה המסופקים על ידי OGSA מאפשרים את הגדרת הגבלות המדיניות האלה אשר חלים ברמת קבוצות או אלמנטים בתוך משאב.

5.5. אינטראקציות עם שאר רכיבי OGSA

החלק הזה מסכם את האינטראקציות בין שירותי המידע ושאר החלקים עם OGSA.

5.5.1.טרנזקציות (Transactions)

שירותי המידע הם דוגמה קלאסית לשירות בו נדרשת תמיכה בטרנזקטיביות, הרבה משאבי מידע (Data resources) מספקים תמיכה עצמאית לטרנזקציות.

ישנם מגוון דרכים בהן טרנזקציות יכולות להיות ממומשות במערכות מבוזרות. טרנזקציות ACID הקונבנציונאליות לדוגמא, או טרנזקציית two-phase-commit, אשר ממש מבצעת סנכרון לבסיס נתונים מבוזר.

בנוסף, תזמון "Time Warp" שירותים יכולים להיות מופעלים ואז לבצע ביטול וחזרה לאחור על הביצוע שלהם כאשר הטרנסקציה מבוטלת.

באופן כללי, הטרנזקציות צריכות להיתמך בכל אינטראקציה של OGSA, לא רק בעבור שירותי מידע. רמת התמיכה לא מתוארת במסמך זה, אבל שירותי המידע יכולים לספק את ההתנהגות שלהם בעבור תמיכה בטרנזקציות.

5.5.2.מעקב (Logging)

בדומה לשאר שירותי OGSA, שירותי המידע משתמשים בשירותי מעקב (Logging Service), לדוגמה לביצוע רישום הפעולות. שירותי המעקב עצמם יכולים להשתמש בשירותי מידע לאחסן את ספרי המעקב (Logs).

5.5.3.שירותי ניהול משימות (Execution Management Services)

לשירותי המידע יש מערכת יחסים קרובה עם שירותי ה-EMS משתמשים בשירותי המידע במטרה להעביר מידע להיכן שהוא נדרש. ההפעלה של שירותי ה-EMS ותכנון ההפעלה צריכים לקחת בחשבון את העלויות של הגישה למידע ממשאב מחשוב מועמד.

5.5.4.תהליכים (Workflow)

תהליכי עבודה יכולים להנחות שירותי מידע לשלוח תוצאות שאילתה לצד שלישי ולבצע המרת מידע כאשר המידע מועתק ממקומו. זה דורש שלתהליך יהיה מידע אינטימי על היכולות של נגישות למידע. בנוסף קריאה לשאילתה מבוזרת יכולה לגרום למספר סוגים של העברת מידע ופעולות על מכונות אחרות. מסיבות אלו שירותי מידע יכולים לספק מידע בעבור מנהל התהליך ולהבטיח שהתהליך יתייחס בצורה מלאה לנושא עלויות ביצוע שאילתות מבוזרות על גבי הרשת ומשאבים נוספים.

5.5.5.ניהול שטח (Provisioning)

בנוסף לשירותי ניהול שטח ומקום אחסון של השירותים עצמם, שירותי מידע גם זקוקים לתמיכה בניהול ותכנון מקום פנוי להעלאת מידע אל המשאבים. חלק עלולים לדרוש תמיכה להעלאת מסננים לשירות מסוים.

5.5.6.הזמנת משאבים מראש (Resource Reservation)

יתכן ששירות מידע יזמין מראש משאב כל שהוא לצורך תפקודו. לדוגמה, העברה של קובץ דורשת מקום אחסון פנוי ורוחב פס ברשת התקשורת, כאשר מערכות שאילתה מבוזרת יכולה לדרוש עוצמת מחשוב נוספת בכדי לבצע פעולות משותפות. בשווה, שירותי מידע חייבים לספק ממשקים אשר מאפשרים להם להיות מוזמנים מראש.

5.5.7.שירותי חיפוש ואיתור (Discovery)

שירותי מידע יכולים להשתמש בשירותי איתור לא רק לשם רישום השירותים עצמם, אלה גם לרישום מערכי המידע אשר שמורים על ידי אותם שירותים. הם יכולים לרשום את מיקום של הגדרת סכמות.

5.5.8.אבטחה (Security)

שירותי אבטחה תומכים במיפוי הזהויות והתפקידים של OGSA למשאב ספציפי. ניהול שירותי VO בדומה מספקים שליטה על המיפויים האלו. שירותי אבטחה גם מספקים תמיכה בעבור בדיקת אמינות והצפנה של העברת מידע.

5.5.9.ניהול הרשת (Network management)

ניהול הרשת הוא נושא קריטי לצורך תכנון העברת כמויות גדולות של נתונים. שירותי המידע יספקו את המידע הנחוץ, וכן את הגירוי הראשוני להתחלת הפעולה, בנוגע לאותם שירותי OGSA המתאימים לקנפג את מאפייני הרשת בכדי להתאים את כמות המידע שנדרש להעביר ואת תלויות הזמן אשר מוגדרות על ידי העברות אלו.

5.5.10. מתן שמות (Naming)

שירותי המידע של OGSA משתמשים בשירות מתן שמות (Naming) של OGSA בכדי לספק שם למערכי המידע. לדוגמה, במערכת קבצים, הכתובת יכולה לציין את מיקומו של קובץ מסוים במכונה מסוימת. השם המופשט יזהה את הקובץ בצורה שאינה תלויה במיקומו הפיזי, ואולי על ידי שימוש בשירות ספריה. השם האנושי יכול להיות שם ידידותי קצר שיכול לקבל את המשמעות המדויקת שלו על ידי התייחסות להקשר בו הוא בא בשימוש.

5.5.11. התרעות (Notification)

התרעות יכולות להיווצר על ידי שירותי ההתראות וזאת במטרה להחצין אירועים מבסיסי נתונים. במסגרת ניהול בסיסי נתונים, לרוב מסופקים מנגנונים אשר מגדירים פעולות אשר יש לבצע כאשר תנאי מסוים מתקיים. כאשר OGSA יכול בצורה קלה לגרום לאירועים אלו להפעיל שירותי התראות. בנוסף, חלק מהמימושים של שירותי התראות יכול לאחסן את מסרי האירועים ובכך לתמוך בשאילתות מצד הלקוח.

6. שירותי ניהול משאבים (Resource Management Services)

6.1. מטרות (Objectives)

שירותי ניהול המשאבים מבצע מספר סוגי פעילויות של ניהול על משאב בGrid. תקן ארכיטקטורת OGSA מגדיר שלושה סוגים של ניהול אשר קשורים למשאבים:

- ניהול של המשאבים עצמם (אתחול, הגדרה של רשת)
- ניהול של משאבים על הGrid (הזמנת משאבים, שליטה ובקרה)
- ניהול של תשתיות OGSA אשר בעצמן מורכבות ממשאבים לדוגמה, ניתור שירות ה Registry

6.2. מודל (Model)

סוגים שונים של ממשקים מממשים סוגים שונים של יכולות ניהול בGrid ה OGSA. ממשקים אלו יכולים להיות מקוטלגים בשלוש רמות, כפי שנראה בעמודה האמצעית בטבלה 8, וכן גם באיור 52.

Type of management	Level of interface	Interface
Management of the resources Themselves	**Resource level**	**CIM/WBEM, SNMP, etc.**
	Infrastructure level	**WSRF, WSDM, etc.**
Resource management on the Grid	**OGSA functions level**	**Functional interfaces**
Management of OGSA Infrastructure		**Specific manageability interfaces**

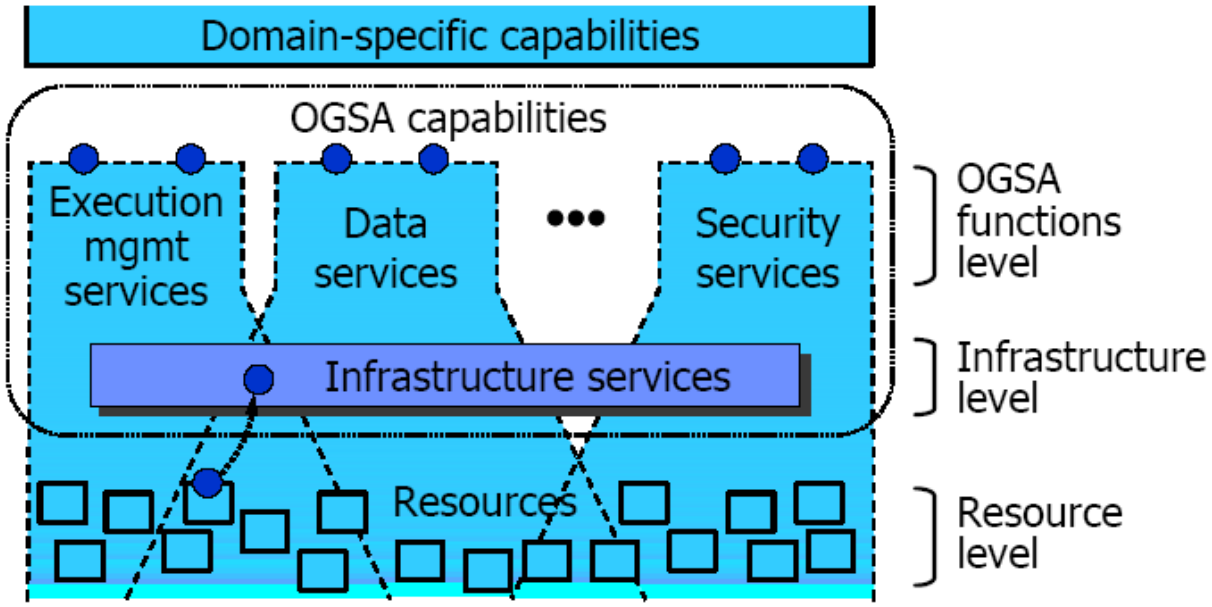
טבלה 8 - יחסיות בין יכולות ניהול לממשקים

אנו מספקים תיאור מפורט של כל סוג ניהול וממשקי הניהול שלו, ראוי לציון שהתיאור מתרכז בממשקי הניהול, ולא במימוש של השירות. בנוסף חשוב לציין ששירות יכול לממש מספר ממשקי ניהול (כאשר ממשקים אלו לא מייצגים ממשקי פונקציונאליות), ובכך שירות הניהול יכול להיות נפרד מהפונקציונאליות אשר אותה הוא מייצג.

ולכן תיאור מבוסס שירות יהיה לא מדויק, ותיאור מבוסס ממשקים נבחר כתחליף. באיור 52, היכולות של OGSA מכסות את כל הרמות, ובעצם מרחיבים את היכולות של המשאבים אשר נדרשים לממש את היכולות של OGSA.

הממשקים האלו מוצגים בעזרת העיגולים הקטנים (ראה איור 52). ברמת המשאב, המשאבים מנוהלים ישירות דרך ממשקי הניהול הפרימיטיביים שלהם (SNMP, CIM/WBEM, JMX, או ממשק ייחודי למשאב כל שהוא). ניהול ברמה הזו כולל ניטור, התקנה, שליטה, ואיתור. משאבים המנוהלים לפי התיאור של מודל המשאבים, אשר מגדיר את המאפיינים שלהם, הפעולות, אירועים, והיחסים ביניהם לבין עצמם. הרמה התשתיתית מספקת את התנהגות הניהול הבסיסית של המשאבים, אשר מבצעת את הפעולות הבסיסיות של ניהול בסביבת OGSA. הסטנדרטיזציה של ההתנהגות הבסיסית הזו דרושה במטרה לקשר את המספר העצום של סוגי המשאבים ומנהלי המשאבים אשר מיוצגים על ידי מספר ספקים. רמת התשתית מספקת:

- מודל ניהול בסיסי, אשר מייצג את המשאבים על-פי תקן WSRF ומאפשר למשאבים ב OGSA לבצע מניפולציה על-פי הסטנדרטים והאמצעים של שירותי הרשת לשם גילוי גישה וכו'. מודל זה מאפשר למשאבים להיות מנוהלים, לפחות ברמה המינימאלית, על ידי האפשרות לגלות לאתר, לנטר ועוד. בכך שמשתמשים במודל בסיסי אחיד של ניהול, נוצרת אחידות בכל רמות הניהול ויתכן שגם מודל משאבים אחיד בנושאים של היחסים, סטאטוס תפעולי או סטטיסטיקות.
- ממשק ניהול גנרי אשר משותף לכל השירותים שממממשים את היכולות של OGSA. לממשקים אלו יש יכולות ניטור ,יצירה, והשמדה של שירותים. ברמת הפונקציונאליות של OGSA, ישנם שני סוגים של ממשקי ניהול, המיוצגים על ידי שני העיגולים מעל היכולות המוצגות באיור 52.
- ממשק פונקציונאלי: חלק מהיכולות המשותפות של OGSA (לדוגמה שירותי EMS) הם סוג של ניהול משאב. שירותים יכולים לספק את היכולות האלה ולחשוף אותם דרך ממשקי הפונקציונאליות לדוגמה יצירה השמדה של ג'וב.



איור 52 - רמות ניהול ב OGSA [15]

- ממשקי ניהול: לכל יכולת יש ממשק ניהול ספציפי דרכן היכולת מנוהלת (לדוגמה ניטור של מנהל ג'ובים). ממשק זה יכול להיות הרחבה של ממשק ניהול גנרי, או הוספה של כל ממשק ניהול שהוא ספציפי ליכולת הניהול.

6.3. יכולות ניהול (Management Capabilities)

ניהול הפונקציונאליות של התשתית מסופק על ידי OASIS WSDM, אשר מתעתד להיות סטנדרד מפתח בנוף ה IT. מגדירי תקן WSDM מפתחים מסמך נפרד המתייחס לניהול של שירותי רשת וניהול בעזרת שירותי רשת. תקן MUWS מספק את מודל הניהול הבסיסי. מתאר כיצד משלבים משאבים לצד ניהול פונקציונאלי בסיסי אשר משותף ל OGSA, כגון ניהול וייצוג המצב (State), הפעולות, והיחסים בין המשאבים. תקן MOWS מספק ממשק ניהול גנרי לניהול השירותים בGrid של OGSA. באשר למודל המשאב, תקן CIM כרגע בהערכה כמודל שיבוא בשימוש ב OGSA. בגלל היכולת שלו להתרחב ולהתרבות, תקן CIM יכול להיות בשימוש בעבור ניהול המשימות על פני מספר יכולות רבות ב OGSA כגון אבטחה, ניהול הפעלה, וניהול עצמי. ברמת הפונקציות של OGSA, יכולת ניהול המשאבים כוללת (אך לא מוגבלת) לניהול מבוזר של פעולות המשאבים ומערכות IT. פעולות אלו יכולות להיות מבוססות מדיניות. לדוגמה לאכוף הגבלות מדיניות אשר הושמו שם במטרה לתמוך בדרישות כגון הזדהות, בחירת פרוטוקול תעבורה, הבטחת רמת שירות, מדיניות פרטיות, וכו'. הפונקציונאליות הכלולה ביכולות ניהול משאבי OGSA כוללת הזמנת משאבים מראש, ניטור ושליטה, ניהול VO, ניהול אבטחת מידע, איתור תקלות וניהול שגיאות, ניהול מדיניות, ניהול קבוצות שירותים ואיתור שירותים, מדידה, התקנה, ואיתור.

6.4. מאפיינים (Properties)

6.4.1.יכולת גידול (Scalability)

ניהול ארכיטקטוני צריך להיות בעל פוטנציאל לגדול לאלפי משאבים. הניהול צריך להתבצע בהיראכיה או משקיף אל משקיף (איחוד\ושיתוף) במטרה להשיג את יכולת הגידול הרצויה, ולכן OGSA מספקת את יכולות הניהול האלו.

6.4.2.קישוריות (Interoperability)

ארכיטקטורת הניהול חייבת להיות מסוגלת לחולל תוכנה, חומרה, ושירותים, גם מעבר לגבולות הארגון ובין המוצרים השונים, כך שסטנדרטיזציה וקישוריות רחבה הם חיוניים.

6.4.3.אבטחה (Security)

ישנם שני אספקטים בניהול אבטחה. ניהול של האבטחה המתייחס לניהול של תשתית האבטחה, כולל הניהול של הזדהות, הרשאות, שליטה בגישה, VO ומדיניות גישה. וניהול מאובטח מתייחס לשימוש

במנגנוני האבטחה למשימות ניהול. ניהול יכול להבטיח את אמינות שלו ולעקוב אחרי מדיניות השליטה בגישה לבעלי המשאבים ו VO.

6.4.4.אמינות (Reliability)

ארכיטקטורת הניהול לא אמורה ליצור נקודת כשל בודדת. לשם האמינות, משאב יכול להיות וירטואלי ומיוצג על ידי מספר שירותים שכל אחד מהם חושף כתובת זהה כנקודה מנוהלת. במצבים כאלו, המערכת אשר מספקת את יכולות הניהול חייבת להיות מודעת לכך ששאילתות על גבי ספק הניהול חייבות להעביר את התוצאות ממספר שירותים אשר מיצרים את הוירטואליזציה של משאב בודד.

6.5. אינטראקציה עם שאר שירותי OGSA

שירותי התשתית מיישמים את הממשקים ברמת התשתית, כגון WSRF ו WSDM. מכיוון שממשקים אלו מגדירים את התנהגות הניהול הבסיסית של המשאבים, כל שירותי OGSA ישתמשו בממשקים אלו אשר מנהלים את המשאבים הללו.

שירותי המידע מספקים פונקציונאליות ניהול משאבים, במיוחד לשם ניטור, מעקב, אשר יכול לבוא בשימוש בעבור איתור תקלות.

שירותי ההפעלה EMS ושירותי המידע הם הצרכנים של פונקציונאליות משאבי הניהול, לשם איתור, תחזית, וניטור של ההפעלה. הם גם ספקים של פונקציונאליות ניהול משאבים ברמת הפונקציונאליות של OGSA.

שירותי ניהול עצמי, באים בשימוש על גבי הממשקים הפונקציונאליים של שירותי ניהול המשאבים וזאת במטרה לשלוט במשאבים (לדוגמה, אפליקציות, התקנים, והמערכים שלהם) על גבי הGrid. הם גם יכולים להיות בשימוש על גבי ממשק ניהול מסוים של שירותי משאבי הניהול, יכולים גם לשלוט בתשתית הGrid עצמו (לדוגמה, ניטור ספריית שירותים, וכאשר הם מלאים ליצר עוד מופעים). אותו דבר תקף גם בשירותי אבטחה, אשר לדוגמה שולטים בגישה אל המשתנים ממשקים פונקציונאליים וגם הממשקים הניהוליים שלהם.

7. שירותי אבטחה (Security Services)

חלק זה מתאר את המטרות המודלים היכולות והמאפיינים של שירותי האבטחה, באינטראקציות עם שאר רכיבי OGSA.

7.1. מטרות (Objectives)

מטרת שירותי האבטחה של OGSA היא לאכוף את מדיניות אבטחת המידע בתוך הארגון הווירטואלי (VO). באופן כללי, המטרה העיקרית היא אכיפת מדיניות האבטחה הארגונית תוך הלימה עם הרמה הגבוהה יותר של מטרות הארגון. לרוב מדובר באיזון עדין, כאשר יש גדילה בעלויות הקשורות לשיפור יכולת האכיפה של המדיניות, כאשר הפוטנציאל לרווחים יכול להיפגע על ידי המורכבות המתגברת וחוסר הגמישות הנובעות מדרישות מדיניות נוקשות.

אפליקציות Grid ספציפיות יכולות לחצות את הגבולות של מספר מרחבי ניהול (Administration Domains) שונים (ראה איור 53) , דבר זה מרמז שכל אחת מהקבוצות היא בעלת מטרות עסקיות שונות, אשר מתורגמות לכל קבוצה באופן אינדיבידואלי. ובכך באופן נפרד נאכפת המדיניות האישית שלהן, אשר משתנה ברמת המורכבות והנוקשות.

יש לשים לב שכל אינטראקציה אשר קשורה במהלך העבודה באפליקציית Grid חייבת להיות מודעת למדיניות המקומית באותה קבוצה וכן למדיניות של הארגון הווירטואלי לדוגמה, הסכמים בין ארגונים, וזאת כדי להשיג את מטרות האבטחה ואת המטרות העסקיות.

מדיניות האבטחה יכולה, לדוגמה, להגדיר את האכיפה של מדיניות לגבי אמינות המסר והסיווג המסר, הזדהות של הישויות הבאות באינטראקציה, הפרדה של סמכויות, מעקב ותיעוד, זיהוי חדירה והוצאה, בדיקות מדיניות הרשאות, מנגנוני שליטה בגישה למשאבים, הבטחת רמת הסביבה, בידוד אפליקטיבי, הימנעות מהתקפות, ושרידות.

רכיבי האבטחה של OGSA חייבים לתמוך, לקשר, ולאחד מודלים נפוצים של אבטחת מידע, מנגנונים, פרוטוקולים, פלטפורמות, וטכנולוגיות בצורה אשר מאפשרת למגוון של מערכות לתקשר בצורה מאובטחת.

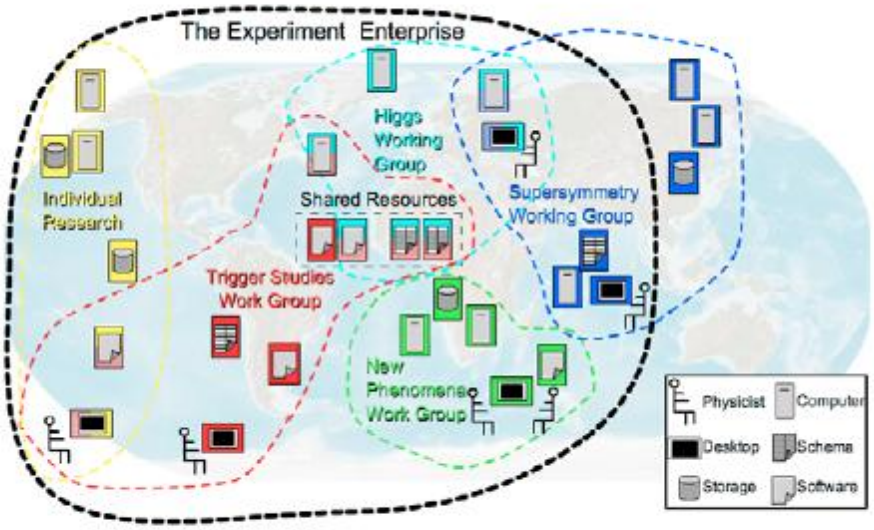
הרכיבים חייבים לאפשר תמיכה לקישוריות עם ארכיטקטורת ומודולי אבטחה קיימים על גבי פלטפורמות וסביבות אירוח שונות. משמעות הדבר שהארכיטקטורה צריכה להיות אדישה לצורת המימוש, ובכך היא יכולה להיות ממומשת בעזרת מנגנוני אבטחה קיימים כגון: Kerberos, PKI וכדומה.

כמו כן הארכיטקטורה אף ניתנת להרחבה, בכך שהיא יכולה לשתף שירותי אבטחה חדשים כאשר הם הופכים להיות זמינים,וכאלה ברי קישוריות עם שירותי אבטחה קיימים. כמו כן, שירותים אשר שוכנים על גבי מספר אתרים וסביבות אירוח צריכים להיות מסוגלים לתקשר אחד עם השני, ולכן קיים הצורך בקישוריות רלוונטי במספר רמות כמו, פרוטוקולים, מדיניות, וזהות.

בנוסף, במקרים מסוימים לא ניתן להגיע ליחסי גומלין מאובטחים בין אתרים לפני הפעלה של אפליקציה. לפעמים בגלל שבאתרים המשתתפים קיימות תשתיות אבטחה שונות (לדוגמה Kerberos או PKI) נדרש להגשים את הקשר המאובטח דרך איחוד של מנגנוני אבטחה שונים.

7.2. מודל (Model)

פרק זה מתאר את המודל התפיסתי של שירותי האבטחה של OGSA. המודל המתואר אינו מודל פורמאלי.



איור 53 - שיתופיות בין ארגונית [15]

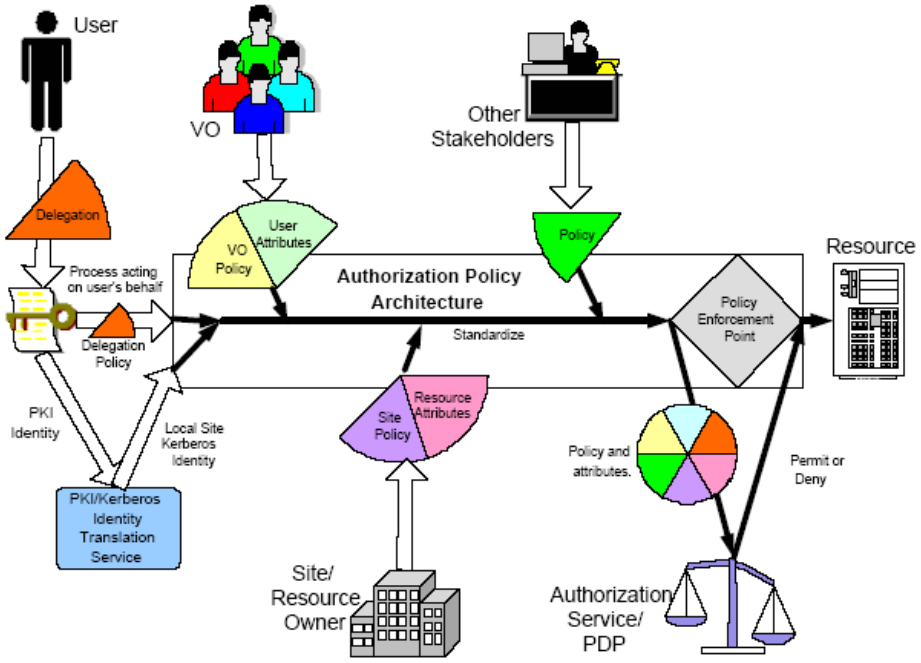
הכוונה הנה לספק שפה שבה ניתן יהיה לתאר ולגבש הבנה משותפת לגבי מדיניות האבטחה אשר נדרש לאכוף במסגרת הארכיטקטורה.

באופן כללי, ניתן לראות שבמסגרת הארכיטקטורה, **ישויות** מבצעות **אינטראקציות** בתוך **הקשר מסוים**. **ישויות** הינן על-פי רוב הן **משתמשים, נושאים או שירותים**. מנגנוני אינטראקציה יכולים להיות כל דרכי התקשורת כגון מייל, טלפון, HTTP, SOAP, SSL, וכו'.

ההקשר (Context) שם את האינטראקציה בפרספקטיבה, ויכול להתייחס לאינטראקציה מקומית המתבצעת במכונה אחת, או יכול להיות מקושר להפעלת שירות בתוך VO, ו\או טרנסקציה מבוזרת. מכלול **הישויות, מנגנוני אינטראקציה וסט ההקשרים** אלו יכולים להיות מתוארים על ידי סדרה של **מאפיינים/תכונות**.

חלק מסוגי התכונות הללו והערכים יכולים לשמש בעבור זיהוי חד ערכי, ואחרים יכולים לשמש בעבור סיווג או קיבוץ. יתרה מכך, חלק מהתכונות הן תכונות שאין מהן מנוס. תכונה שאין ממנה מנוס יכולה לזהות ישות בעצמה בלי שום מצביע לסמכות חיצונית. דוגמה לתכונה שכזו אשר מהווה תכונה משותפת היא צמד מפתחות פרטי \ ציבורי.

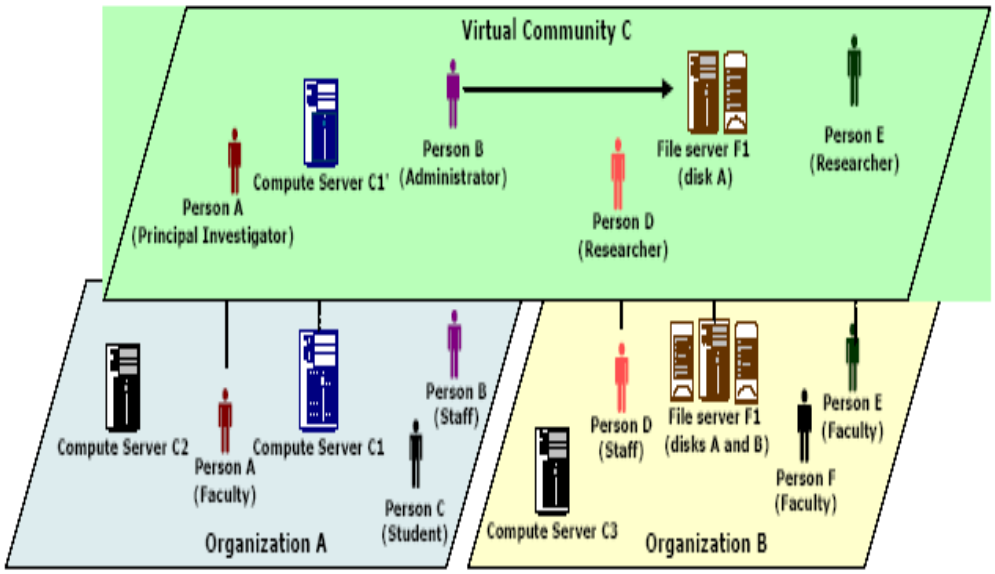
מדיניות אבטחה (Security policies) הנה הצהרות לגבי הישויות, מנגנוני האינטראקציה , ההקשרים וההגדרה של ההגבלות הקשורות לערכי המאפיינים. ניתן יהיה לבטא את הצהרות המדיניות (Security policies) במושגים של הישויות (Identities & User Properties), משאבים (Endpoints), ומאפייני הסביבות (Time , Location, Purpose, Requester trust level or Request path). הצהרות מדיניות אלו יהיו באספקטים הכוללים הרשאות, הזדהות, זהות, מיפוי וכו'.



איור 54 - דוגמא לקלט עבור מדיניות החלטה ואכיפה [15]

המודל מגדיר את שירותי אבטחה כישויות בעלי דפוסי אינטראקציות אשר מסייעות לניהול, ביטוי, פרסום, גילוי, התקשורת, אישוש, אכיפה והתאמה של מדיניות האבטחה.
במילים אחרות מדיניות אכיפת האבטחה היא המטרה האולטימטיבית, ושירותי האבטחה מתוכננים ופרוסים במטרה לתמוך בה.
בפירוט מדויק של המודל, אנו נזהה מספר ישויות, מנגנוני אינטראקציה, והקשרים, בפרק זה אנו נדון על חלק מהמאפיינים והיחסים המשותפים.

איור 54 - מראה דוגמה של אכיפת מדיניות אשר מגנה על השימוש במשאב. קונספטואלית המודל המוצג לא שונה ממודל שירותי רשת באופן כללי ועוד ארכיטקטורות מחשוב מבוזרות. מערכות Grid שמות דגש ברור על הישויות והדפוסים הקשורים לארגונים ומעבר להם, ושירותי האבטחה אשר מאפשרים את האינטראקציות הללו, ניתן לומר שאינטראקציות אלו גם קיימות באפליקציות עסקיות היכן שיש אינטראקציות בין עסק לעסק.
איור 55 - מראה שני ארגונים וקהילה וירטואלית אשר נמשלות על ידי המדיניות של עצמן. ניתן לציין שהישויות בתוך אותה קהילה וירטואלית הן בעלות מאפיינים ותכונות, אשר שונים ואין להם קשר לאלו שנמצאים בקבוצה הביתית שלהם. זה מדגיש את העובדה המרכזית כי מודל שירותי האבטחה צריך לתמוך באכיפה במקביל של מספר policies, שכל אחת מהן נבחנת ב context שלה פרטי.



איור 55 - רעיון הקהילה הוירטואלית [15]

7.3. תסריטי דוגמה (Example Scenarios)

7.3.1.תסריט ספריה דיגיטאלית (Digital Library)

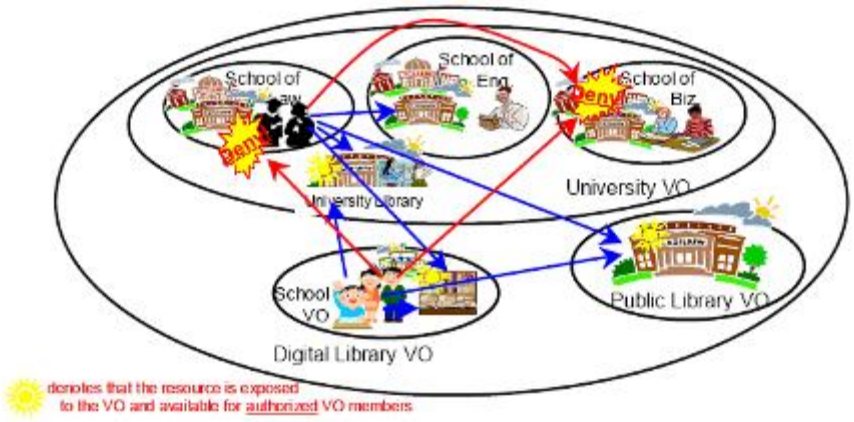
תסריט זה מתייחס לספריה דיגיטאלית המופעלת על ידי ארגון ציבורי. בתסריט זה ,מספר נתון של בתי ספר וספריות ציבוריות משתתפים בתכנית הספרייה הדיגיטאלית. במסגרת פעילות בתי הספר מורים וסטודנטים הפעילים בהם הנם הבעלים של אמצעים לשיתוף חומר חינוכי כמו ספרים דיגיטאליים, סרטים, תמונות ועוד חומרים דיגיטאליים המשמשים למטרות חינוכיות אשר בדרך כלל נשמרו בבתי ספר אינדיבידואלים או ספריות.

כל אחד מהמשתתפים אחראי למשתמשים הספציפיים שלו ולמשאבים שלו (חומרים לשם חינוך) ולניהולם, כגון רישום או מחיקה של משתמשים ומשאבים. ישנם מיקרים אשר בהם בתי ספר, לדוגמה אוניברסיטאות, מורכבים ממספר תתי-גופים. במקרים כאלו כל תת ארגון באוניברסיטה יכול להיות VO בעצמו והאוניברסיטה עצמה יכולה להפוך להיות הצרוף של ה VO הללו.

התסריט הזה מתואר באיור 56.

להלן כמה דוגמאות, להצהרות מדיניות אבטחת מידע רלוונטיות לתסריט המתואר.

- כל הסטודנטים המשויכים לבית ספר מסוים הנם בעלי גישה לקריאת החומרים בעבור סטודנטים.
- כל המורים הם בעלי גישת קריאה לחומרים גם של סטודנטים וגם של הפקולטות.
- חלק מהמורים אשר מורשים על ידי בית הספר יכולים להוסיף חומרים חדשים.

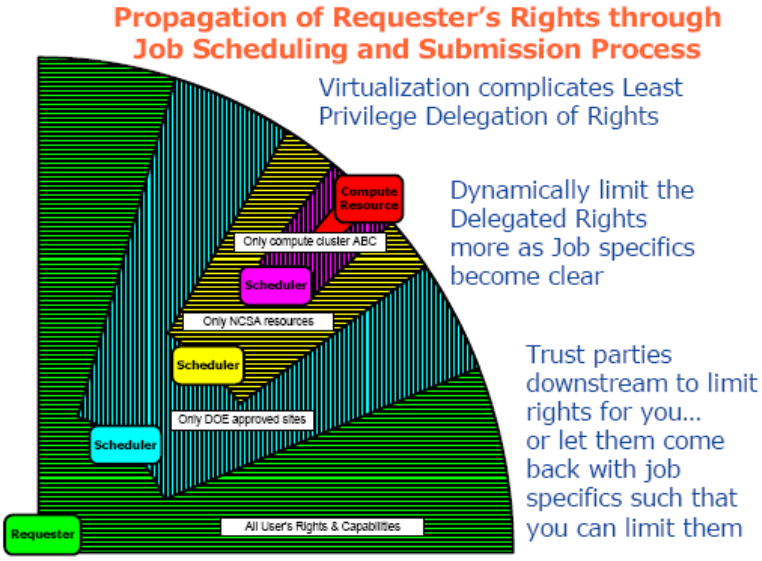


איור 56 - ספריה דיגיטאלית כדוגמא לקהילה וירטואלית [15]

ספרית בית הספר באוניברסיטה יכולה לאפשר גישה מתוך האוניברסיטה, אבל לא תאפשר גישה מהספרייה הדיגיטאלית VO.

7.3.2.תסריט ייצוג הרשאות מינימאלי (Least Privilege Delegation)

ייצוג הרשאות הוא יכולת בסיסית אשר נדרשת בכדי לאפשר לשירותים לעבוד בשם ישויות אחרות. בעקבות הרשות לייצוג הרשאות ישנו סיכון שכל אחד מהשירותים האלה עלול להתפשר ולהשתמש בהרשאות אלו בדרכים לא נאותות. במטרה להפחית את החשיפה, היינו רוצים להפחית את ייצוג ההרשאות רק לאלו אשר באמת זקוקים לשירות. מודל ייצוג ההרשאות המינימאלי דורש את היכולת להתאים בין פעולות שמבצע השירות המופעל למדיניות המתאימה – כמות הרשאות הנכונה.



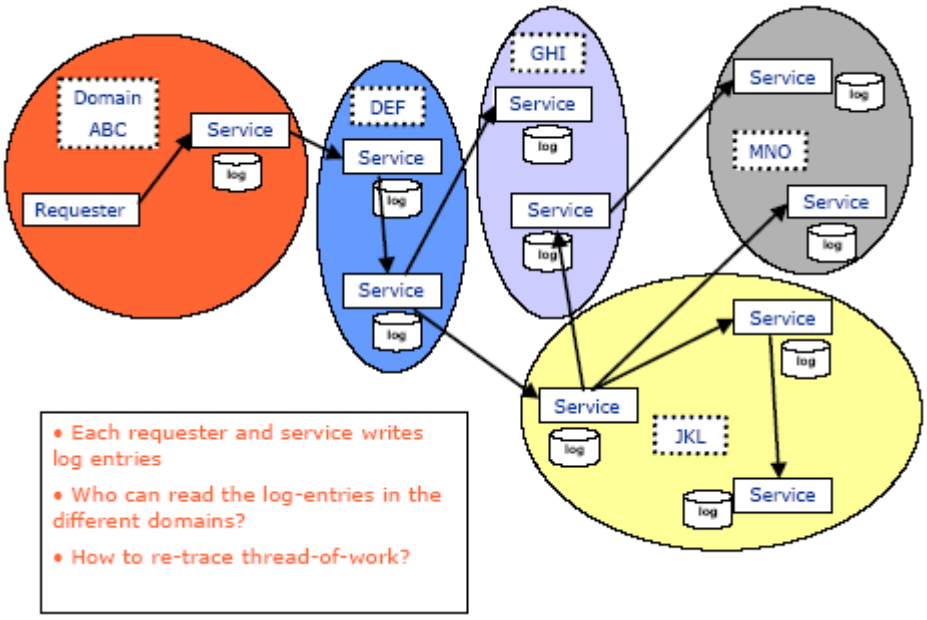
איור 57 - פריסת הרשאות [15]

הרבה מערכות Grid משתמשות ברעיון המשימה (Job), כאשר תיאור המשימה מוגדר בשפה שלהן. דרישות הג'וב מושוות עם היכולות והזמינות של המשאבים על ידי איתור, מנגנון ברוקר, ושירותי

תזמון. השפה שמשתמשים בה במטרה לבטא את הנחיות המשימות (Jobs) ויכולות המשאבים צריכה להיות מסוגלת להשוות עם ההנחיות אשר מבטאות את ההרשאות הדרושות. כל חריגה כנראה תגרום לפריסה של יותר מידי הרשאות לשירות כדי להבטיח שהנחיות הג'וב יבוצעו, כפי שמודגם באיור 57.

7.3.3. תסריט מעקב מאובטח בסביבה מבוזרת (Secure logging in a distributed environment)

שירותי מעקב, וגישה מאובטחת ללוג למטרות חשבונאות, הופכות לבעיה קשה יותר בסביבה מבוזרת, מכיוון שהלוג של שירות יכול לשכון במקום מרוחק (כפי שמוצג באיור 58).
שירותי לוג צריכים להיות בעלי מאפייני אבטחה כך שהלוג יהיה מאובטח, הם צריכים להיות חסינים לנזקים, ולהבטיח את אמינות המסרים. כאשר אנו מקבלים את רמת התיחום הזו, היא מובילה למעקב אחר אירועים מוקלטים בצורה מאובטחת. אילו יכולים להיות אירועי אבטחה, אירועים עסקיים או טרנזקציות.



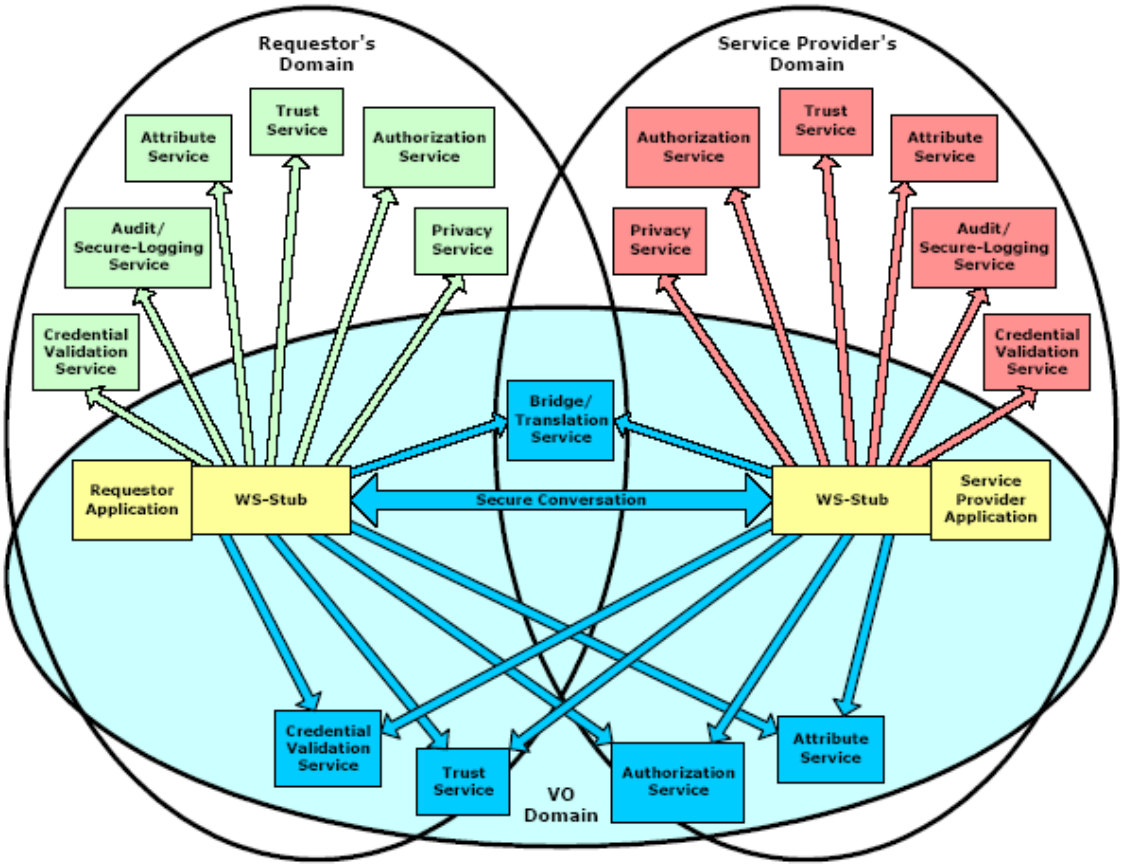
איור 58 - שירותי לוג מאובטח בסביבה מבוזרת [15]

7.4. יכולות פונקציונאליות (Functional Capabilities)

בחלק זה אנו מתארים את הפונקציונאליות של יכולות האבטחה בארכיטקטורת OGSA וכיצד הן מתקשרות לשירותי האבטחה ולדפוסי השימוש שלהם. תיאור מפורט יותר של השירותים ודפוסי השימוש בהם יסופק בהמשך.
איור 59 ממחיש כיצד גם הלקוח וגם ספק השירות משתמשים בתשתיות אבטחה שונים בכדי להבטיח את תאימות המדיניות. יש לשים לב שבתמונה הקריאה מבוצעת מתוך ה-stub, ומחוצה לו וכן בצורה שקופה לאפליקציה.
הציפייה הנה שאכיפת המדיניות תתבצע ותטופל בצורה כזו, שכאשר יש את המוטיבציה ליהנות מהיתרונות של תחזוקת קוד ספציפי למינימום מפתחי אפליקציות. יתרה מכך איור 59 מראה בצורה ברורה שבכדי לבצע קריאות לשירותים בהתאם למדיניות ה VO, הקריאות מתבצעות למופעי שירותי

אבטחה שונים אשר מנוהלים בארגונים שונים. להלן היכולות הפונקציונאליות ושירותי האבטחה בהתאם:

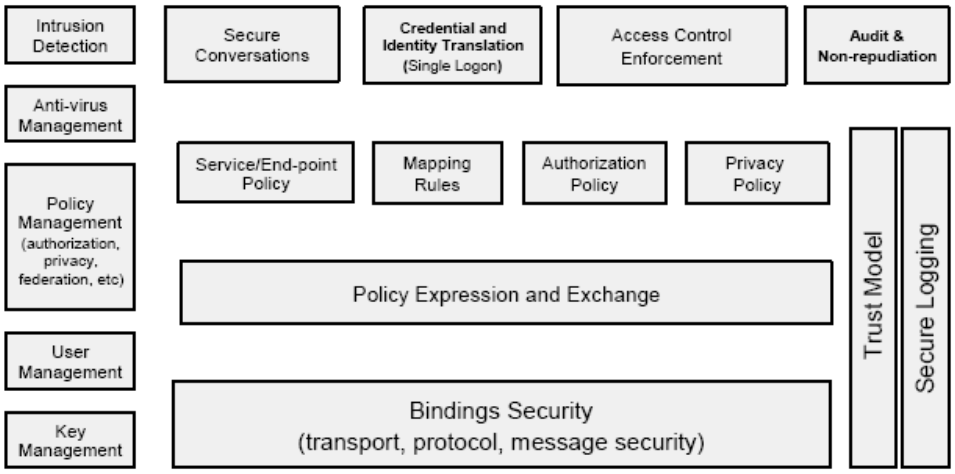
- **שירות הזדהות (Authentication)** - ההזדהות מתייחסת לזיהוי נטול טעויות. הפונקציונאליות היא חלק מאישוש מאפייני הזיהוי ושירותי Trust באיור 59. דוגמה אחת היא זיהוי המשתמש על-פי צירוף שם משתמש וסיסמה, דוגמה נוספת אשר מערבת זיהוי משתמש דרך מנגנוני Kerberos, וכרטיס אשר מועבר לסביבת האירוח של מבקש השירות אשר קובע את אמינות זהותו של הכרטיס לפני שהשירות מאותחל.
- **שירות מיפוי הזהות (Identity mapping)** - שרות הגישור/תרגום למאפייני הישויות ול - Trust (איור 59) מספק את יכולות המרת זהות אשר קיימות בקבוצה אחת לזהות בקבוצה אחרת. לדוגמה, נעריך זהות משתמש בצורה של X.500 חד ערכי, אשר מוכל בתוך תעודה דיגיטאלית X.509 V3. הצירוף, של הנושא DN, המפיק של התעודה והמספר הסידורי של התעודה יכול להיחשב כמפתח לזהות של מבקש השירות. התיחום של זהות הקבוצה בדוגמה זו יכול להיחשב כסדרה של תעודות אשר הופקו על ידי CA. בהנחה שהשימוש בתעודה הוא לשם אישוש זהותו של מבקש השירות, מיפוי זהותו של המשתמש מתבצע על ידי שירות מיפוי הזהות של המבקש.



איור 59 - שירותי אבטחה בסביבה מבוזרת [15]

- **שירות הרשאות (Authorization)** - שירות ההרשאות מתייחס למימוש שליטה בגישה מבוססת החלטות מדיניות. שירות ההרשאות מקבל קלט אשר כולל בתוכו את הזהות של מבקש השירות, ובהתאם לזהותו של המשאב וההגבלות מבוססות מדיניות, מתקבלת ההחלטה האם מבקש השירות מורשה לגשת למשאב. ניתן לצפות שסביבת האירוח התואמת ל OGSA תספק את פונקציות שליטה על הגישה, ואת שירות ההרשאות בהסתמך על סוג הגישה אשר נאכפת.
- ***שירות המרת זהות (Credential conversion)*** - שרות הגישור/תרגום למאפייני הישויות ול - Trust (איור 59) מספק את יכולות המרת ***Credential*** מסוג אחד לסוג אחר או מצורה אחת לצורה אחרת. *דבר* זה יכול לכלול משימות כמו הערכה מחדש של חברות בקבוצה, הרשאות, מאפיינים וחריגות הקשורות בישויות. לדוגמה,שירות *Credential conversion* יכול להמיר *Credential* Kerberos לצורה בה הוא נדרש על ידי שירות ההרשאות. שירות Credential Conversion מבוסס המדיניות מנצל את הקישוריות של סוגי credentials שונים, אשר יכולים להיות נצרכים על ידי שירותים. ניתן לצפות ששירות CC ישתמש בשירותי מיפוי הזהות.
- ***רישום מאובטח (Audit and Secure logging)*** - שירותי הרישום, בדומה למיפוי הזהויות וההרשאות, גם הם מונחי מדיניות. שירותי הרישום אחראים להפיק רשומות אשר עוקבות אחרי אירועי אבטחה רלוונטיים. הפחתה של רישום התוצאות יכולה להבחן כך שמדיניות האבטחה הרצויה נאכפת.
- **שירותי פרטיות (Privacy)** - שירות ה privacy במהותו מבוסס על מדיניות זיהוי מידע אישי. צרכני השירותים וספקי השירותים יכולים לאחסן מידע מותאם אישית בעזרת שירותי הפרטיות. שירות שכזה יכול להיות בשימוש במטרה לאכוף מדיניות VO. מבט שונה על היחסים בין רכיבי האבטחה הרלוונטיים המוצגים בציור כמחסנית שכבות של השירותים הקשורים.

איור 60 מספק מבט על לגבי אוסף רכיבי אבטחת המידע והקשרים בניהם בארכיטקטורת OGSA . כלל ממשקי האבטחה אשר בשימוש על ידי מבקש השירות וספק השירות צריכים להיות סטנדרטיים בהתאם ל-OGSA. מימוש מותאם יוכל להשתמש בשירותים ומדיניות שהוגדרה דרך קונפיגורציה. מימוש תואם OGSA של ממשק אבטחה ספציפי יכול לספק שירות אבטחה אלטרנטיבי.



איור 60 - מיפוי הרכיבים במודל אבטחה של Grid [15]

7.5. מאפיינים – (Properties)

באופן כללי המאפיינים של שירותי האבטחה תלויים בדרישות הטכנולוגיות אשר נובעות מהמדיניות אשר יש לאכוף. לדוגמה, במטרה לאפשר אכיפת מדיניות, רמת שירות מסוימת חייבת להיות ממומשת על ידי שירות האבטחה, אשר מתרגם את המאפיינים כמו מקסימום זמן תגובה, latency, זמינות, והתאוששות. במקרים רבים, המימוש של שירותי אבטחה יוכל לאכוף את המאפיינים המבוקשים דרך השימוש בשירותים אחרים. לדוגמה, שירותי המידע יכולים להשתמש במאפייני השירות על מנת לגשת למידע על החריגות מדיניות אבטחה ב LDAP או RDBMS, ויכול להשתמש ביכולות שיכפול המידע לשירותים אלו במטרה להשיג את הזמינות הנדרשת במטרה לאכוף מדיניות אבטחה.

7.6. אינטראקציות עם שירותי OGSA (Interactions with other OGSA services)

באופן כללי, כלל ההפעלות של שירותי OGSA נתונים לאכיפה וההגבלות של מדיניות האבטחה הרלוונטית. במקרים מסוימים, אכיפה זו ברורה ומקודדת, ובמקרים אחרים היכולת לחבר אותה בקלות או לקרוא לה משירותי תשתית אבטחה חיצונית היא חיונית.

במבט כזה, כל שירותי OGSA תלויים על השכבות העליונות של שירותי האבטחה. ובמקרים אחרים, שירותי האבטחה והדרישות מקושרות בצורה אינטימית לשאר שירותי OGSA או רמה גבוהה יותר. לדוגמה, ניתן באמצעות מאפיין של שירות לאפשר לשירות מידע לאחזר מדיניות הקשורה למידע מה registry או בסיס נתונים. לפי כך, שירותי אבטחת המידע יכולים להיות נצרכים על ידי שאר שירותי OGSA.

8. שירותים בניהול עצמי (Self-Management Services)

8.1. מטרות (Objectives)

ניהול-עצמי (Self Management) נתפס כדרך להפחית את המורכבות הקשורה בעלות הבעלות והתפעול של תשתיות ה-IT (TCO). בסביבה מנוהלת בצורה עצמית, רכיבי המערכות כולל חומרה, מחשבים, רשתות, התקני אחסון, ורכיבי תוכנה כגון מערכות הפעלה ואפליקציות עסקיות – הנם בעלי מאפיינים ותכונות של: self-healing, self-configuring, self-optimizing אשר מהווים את המאפיינים המרכזיים של ניהול עצמי, כפי שמתואר בסעיף 8.4.

מאפיינים אלו מרמזים שהמשימות הקשורות לקינפוג, התאוששות ואופטימיזציה של מערכות IT יכולים להיות מופעלים בהתבסס על המצב שהרכיב עצמו מגלה, מונע על ידי צרכים עסקיים, ומשימות אלו מבוצעות על ידי אותם הטכנולוגיות.

באופן מרוכז, יוזמות ומאפיינים אלו מאפשרים לארגון לתפקד בצורה יעילה עם כוח אדם מועט כאשר העלויות מפחתות ומוגברת יכולת הארגון להגיב לשינויים. לשם הדוגמה במערכות מנוהלות בצורה עצמית, משאב חדש פשוט מותקן ואז האופטימיזציה מתבצעת. דבר זה משדרג בצורה משמעותית את המימושים המסורתיים, אשר זקוקים לניתוח רב לפני ההתקנה, על מנת להבטיח שהמשאב ירוץ בצורה יעילה.

למרות העובדה כי מצופה שמערכות מנוהלות-עצמית יציגו את מכלול או חלקם של מאפיינים אלו, הרי שמאפיינים אלו הם חלק מהטבע האוטונומי של המערכת ככלל. אחת מהמטרות העיקריות של מערכת לניהול עצמי היא לתמוך באכיפת רמת שירות על מערך של שירותים (או משאבים, בהתאם לנושא) - עם אוטומציה מקסימאלית ככל שניתן, וזאת במטרה להפחית אל העלויות והמורכבות לניהול מערכות.

בסביבה מבצעית, נדרש לשלוט במגוון אספקטים של התנהגות של הרכיבים המיישמים את הפתרון בצורה אשר יכולה להיקבע מראש על ידי מפתח הרכיב. במערכות בעלות יכולת ניהול עצמית, ניתן להשיג מטרה זו על ידי שימוש ברכיבי SLM (Service Level Managers) רכיבי ה SLM אחראיים להגדרה והתאמה של מדיניות, שינוי ההתנהגות של משאבים מנוהלים או שירותים שאחראיים לצפות תנאים אשר מבטיחים את התאימות הכוללת של מטרות הארגון.

רכיבי ה SLM עצמם מנוהלים על ידי מדיניות אשר מוטמעת במימוש שלהם או מתקבלת משירותי SLM אחרים. ולכן יתכן ששירות זה יהיה בעל אספקטים של פעילויות ניהול עצמי. ולכן קיימת היראכיה בין שירותי SLM, אשר מפחיתים בצורה משמעותית את המורכבות של הפעולות ההתחלתיות של עיצוב מערכות, מכיוון ש SLM יכול להיות בנוי על-פי SLM פשוט בתהליך.

8.2. מאפיינים בסיסיים (Basic Attributes)

הקיבוץ של מאפיינים נדרש במספר שלבים של ניהול עצמי כפי שמתואר למטה.

8.2.1. חוזי רמת שירות (Service Level Agreement)

ה-SLA כולל הסכמים עסקיים או IT בין ספק השירות והמשתמשים של השירות. ה SLA מספק הנחיה שמתבטאת במושגים הניתנים למדידה, באשר למטרות של השירות או משאב מנוהל.

8.2.2. מדיניות (Policy)

משתמשים במדיניות לאכיפת מדיניות של SLM וניהול משאבים תחת השליטה שלו. מדיניות השולטת בהתנהגות של ישויות מנוהלות עצמאית נגזרת מה SLA, מוגדרת כחלק מהקשר לניהול ותחת המשאבים המנוהלים בשימוש SLM ומתזמרים בזמן אמת את השינויים בתשתית ניהול דינאמית, מבוססת על מדיניות אשר שולטת בהתנהגות שלו.

8.2.3. מנהל מודל רמת השירות (Service Level Manager Model)

המודל הזה מספק תשתית לאתחל ולעבוד עם SLM כאשר מגוון ה SLMs והמתפעלים האנושיים יכולים לבצע אינטראקציה אחד עם השני מבלי להיות בעלי ידע אחד על השני בזמן העיצוב. קיים רכיב התממשקות למודל ה SLM, וכן לייצוג של SLM ורכיבי ניהול השירותים ומעגלי השליטה אשר הם מייצרים. הדפוס של מעגלי השליטה מתחלק ל ניתור, ניתוח, תכנון והפעלה.

8.3. תסריטי דוגמה (Example Scenarios)

יכולות הניהול הן בסיסיות לGrid. חלק זה מתאר שתי דוגמאות לשימוש בעולם האמיתי: ניהול רמת המשימות (Jobs) ורמת ניהול של הGrid, רעיונות דומים יכולים להיגזר מניתוח כיצד ניהול עצמי עובד בדוגמאות אחרות. לדוגמה אבטחה וכדומה.

ניהול רמת המשימות (*Job Level management*)

מנהל הפעילות משגר פעילות למשימה (Job) ומנהל את המשא ומתן על SLA של המשימה. המשימה חייבת להיות מופעלת בדרך אשר מרצה את ההסכם. בשלב מאוחר יותר מנהל פעילות ה IT יכול לבחור לנהל את המשא ומתן מחדש ולהתייחס לדרישות עסקיות חדשות. דבר זה מתבצע מול מנהל המשימות המעדכן הסכם קיים.

כאשר ההסכם מעודכן, דרישות המשאב מחושבות מחדש במעגל רמת השירות (ניתוח והקרנה), שלבי התחזית (כולל הקצאת משאבים והתקנה) הם פעולות שמופעלות כתוצאה של שינוי תנאים. לאחר שלבי התחזית המשאבים מוכנים כרכיבים הדרושים למשימה על מנת להתחיל.

כל זאת, כולל הפעלה של קבצי הרצת משאבים כגון Application Server או DBMS. בשלב זה מנהל פעילות ה IT יכול לנטר את העומס על המשאבים והניצולת שלהם, בעזרת שירות הניטור והמדידה. בשלב זה , מנהל פעילות IT גם יכול לקבל את הסטאטוס של משימה רצה ממנהל המשימות.

8.3.1. ניהול רמת מערכת הGrid (Grid System Level management)

ברמת מערכת הGrid, המטרה המרכזית הנה לשפר את ניצולת המשאבים תוך שמירה על ה SLA של משימות אשר בביצוע. מנהל מערכת ברמת הGrid יכול להוסיף משאב חדש כאשר הוא מתכונן לעומס מתגבר וכן לשחרר משאבים במטרה להפחית עלויות.

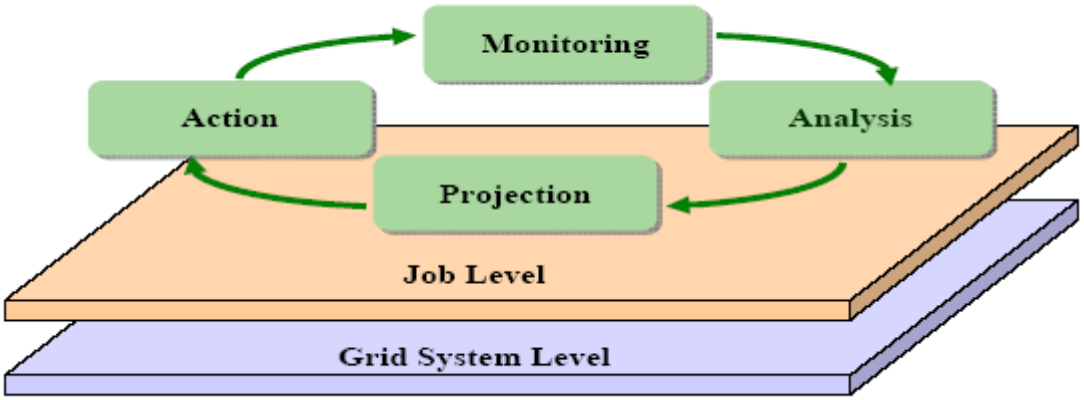
המשאבים המוקצים לטובת משימה יכולים להיות מותאמים בהתאם למדיניות, לדוגמה, עדיפות של משימה אחת ביחס למשימה אחרת במערכת הGrid. בשלב האנאליזה וההקרנה, מוערכים המשאבים הזמינים והעומס הנכון, כמו כן ניצולת ועומס העתידי. הגברת הניצולת יכולה להפעיל פעולה אשר תוסיף משאב למערכת הGrid מתוך מאגר המשאבים הזמינים. לדוגמה על ידי הזזה של המשאבים ממערכות אחרות או על ידי שחרור משאבים אשר נמצאים בעדיפות נמוכה.

8.4. יכולות פונקציונאליות (Functional Capabilities)

יכולות ניהול עצמי מתייחסות לקינפוג עצמי, התאוששות עצמית, ואופטימיזציה עצמית. זה מראה את ההבדלים של ניהול עצמי משאר קטגוריות OGSA: מעגלי שליטה נוצרים ומערכות מתנהגות בצורה אינטליגנטית, בהתבסס על שינויי הסביבה. המנגנון אשר יוצר ניהול עצמי יכול להיות מתואר בצורה הבאה:

קינפוג עצמי , מנגנון אשר מאמץ בצורה דינאמית שינויים בסביבת ה IT, מבוסס מדיניות המסופקת עלידי מומחי ה IT, בעזרת המדיניות הזו שינויים יכולים לעורר בקשות אשר מובילות, לדוגמה, להתקנה של רכיב חדש או הורדה של קיים, וזה יכול בצורה משמעותית להגביר או להפחית את עומס העבודה.

מנגנון ריפוי עצמי יכול לחוש בפעילות לא תקינה של משאב ושירות ולהתחיל פעילות מבוססת מדיניות מבלי להפריע לסביבת ה IT. ליכולת ההתאוששות העצמית יש אלמנט של הגנה עצמית הכלולה בו גם כן וזה אומר שלרכיבים יש את היכולת לחוש בהתנהגויות עוינות כאשר הן מתרחשות ולנקוט בצעדים לתיקון המצב בכדי להפוך אותם לפחות פגיעים.



איור 61 - דוגמא לתרחישים: שכבת הGrid ושכבת ה Job [15]

ההתנהגות העוינת יכולה לכלול גישה לא מורשית או שימוש לא מורשה, התקפת וירוס או התקפה לשם מניעת שירות.

מנגנון לאופטימיזציה עצמית גם יכול לכוונן את עצמם ליעילות מקסימאלית וזאת במטרה לעמוד בדרישות הארגון. פעולת הכוונון יכולה להתבטא בהקצאה מחודשת של משאבים במטרה לשפר את הנצולת הכוללת או אופטימיזציה על ידי אכיפה של SLA. מנגנון אופטימיזציה עצמית משתמש ביכולת הקינפוג העצמית לשם מימושו. חשוב לציין שמנגנונים אלו הם קשורים אחד לשני. הם עובדים ביחד במטרה לאפשר את השינויים והקונפיגורציה של מערכות ה IT. כל הקטגוריות של שירותי OGSA מכוונים להשיג יכולות ניהול עצמי.

8.4.1.ניהול רמת השירות (Service level management)

ניהול רמת השירות מבטיח שה- QoS נשמר. שמירת רמת השירות מתבצעת דרך הפעילויות המתוארות ביתר פירוט בפסקה 8.1.

- **ניטור (Monitoring):** שירות ה-SLM מקבל ומעבד מידע לגבי המשאבים דרך רכיב ניטור. מידע לגבי הפעלת השירותים וניטור צריכת המשאבים – לדוגמה, ניטור העומסים והשימוש במשאבים וכן מצב רכיבי השירותים, זיהוי תקלות, יכול להתקבל בעזרת שירות הניטור. שירות מידע שכזה בא בשימוש כקלט בשלב הניתוח של המידע.
- **ניתוח והשלכה (Analysis and Projection):** הניתוח מתבצע בתהליך שבו נדרש להעריך ולהחליט האם מתקיימת תאימות בין המדיניות וה-SLAs. המנהל יודע האם משאבים עומדים במשימות QoS ופועלים לפי המדיניות המותרת. תהליך הניתוח יכול לחזות התנהגות משאבים עתידית בהתבסס על התנהגות היסטורית והדרישות המוקרנות עליהם. כאשר התנהגות המערכת אינה אחידה ואינה תואמת את המטרה הכוללת, המנהל מבצע הערכת אלטרנטיבות של תסריטים ודרכי פעולה אפשריות במטרה לנסות ולהשפיע על המשאבים ולאחר מכן בוחר דרך פעולה התואמת את המדיניות שלו.
- **פעולה (Action):** המנהל מוציא לפועל את התכנית\פעולה דרך אינטראקציה עם המשאבים המנוהלים או דרך התקשרות עם מנהלים אחרים אשר אחראיים על אספקטים נוספים במערכת. לדוגמה, מנהל עומסי העבודה מתאים את סדרי העדיפות ואת חלוקת העבודה בין המשאבים במטרה לעמוד במטרות רמת השירות והמדיניות. או, במקרים שבהם פעולה מקומית לא מתאפשרת (כתוצאה מאי קיום מדיניות הקשורה בפעולה) או שהפעולה אינה יעילה, מנהל המשאבים יכול לפנות לפונקציות ניהול נוספות במטרה להשתלט על המצב, לדוגמה מנהל עומסי העבודה יכול לדרוש עוד רכיבי עיבוד.

מעגל השליטה המתואר כאן מבצע כל הזמן הערכה של מצב המערכת אל מול מטרות רמת השירות הנדרשות, ואז המעגל מבצע את ההתאמות הנדרשות במערכת. בכדי שפעולות אלו יסתיימו בהצלחה שירותים נוספים נדרשים- לדוגמה capacity planning, entitlement of resources, problem and root cause analysis, predictive analysis, וכדומה.

מספר רב של רכיבי ניהול יכולים להיות מעורבים בפעילות שמירת רמת השירות. דרישות ה-QoS וה-SLA שמקבלות התייחסות בפעילויות ניהול אלו יכולות להיות מאוד מגוונות. לדוגמה, הו יכולות לכלול

אכיפת יעדי ביצועים או צריכה (ניהול עומסי עבודה), או עוד מטרות כגון רמת אבטחה. פעילויות אכיפת רמת השירות משפיעות באופן ישיר ועקיף אחת על השנייה, ולכן חייב להישמר צימוד רופף בין מנהלי רמת השירות.

8.4.2.מדיניות וניהול מבוסס מודל (Policy and Model based Management)

מנהל רמת השירות מתזמר בזמן אמת שינויים בתשתית הניהול הדינאמית בהתבסס על מדיניות אשר מכתיבה את ההתנהגות שלהם. החוכמה של ניהול עצמי מוכתבת על ידי המדיניות והמודל לגבי ה SLMs אשר מנוהלים.

8.4.3.אישור קבלת זכויות (Entitlement)

חלק זה דן במשא ומתן עם עוד בעלי משאבים (SLMs מתפעלים אנושיים, ומאגרי משאבים). במטרה להשיג את המשאב הנכון. במהלך תהליך זה SLA שונים מופעלים בכדי למצוא איזה צורך של משאב יותר חשוב. האישור הוא שלב מוקדם יותר לזימון המשאב ומספק צימוד רופף יותר, אופציה לזימון. בניגוד, הזמנה מראש מבטיחה גישה למשאב.

8.4.4.תכנון (Planning)

תכנון מתייחס לחישוב הדרישות האופציונאליות של השירות במושגים של המשאב אותו הוא דורש. זה יכול להיות בשלב האתחול, בעקבות בעיה שזוהתה, או פקודה מרמה גבוהה יותר.

8.4.5.ניהול קיבולת (Capacity Management)

ניהול הקיבולת כולל את הפעולות המתייחסות לעדכון המצב הנוכחי והדרישות של המשאב. זה כולל נושאים כמו קישור למלאי, ניהול הנכסים, הזזה של משאבים מהמיקום הנוכחי לקבוצת שירותים, הזזה של משאבים מקיבוץ שירותים אל מאגר או לקבוצה אחרת.

8.4.6.יכולת לחזות (Provisioning)

היכולת לחזות, כוללת את תתי-הפעולות של התקנה וקינפוג, אלו פעולות חשובות אשר מתבצעות במסגרת התמיכה בניהול עצמי כאשר משאבים מתכוננים לקראת השימוש העתידי בהם. היכולת לחזות נתמכת על ידי מספר שירותי OGSA, כגון שירותי תוכן אפליקטיביים אשר מתחזקים את תיאורי ההתקנה והקינפוג.

8.4.7.ניתוח (Analytics)

תקלות וניתוח מקור הבעיה. ניתוח מידע מנוטר במטרה למצוא את מקור הבעיה בזמן זיהוי תקלה, כולל יכולות סינון או קורלציה ועוד. ניתוח אחר הוא של מידע מנוטר במטרה לחזות התנהגות עתידית ולתכנן את צורכי המשאב או לחזות תקלות עתידיות.

8.5. מאפיינים (Properties)

רכיבי ניהול רמת השירות מממשים את יכולות הניהול-העצמי ויכולות מבוססות מדיניות על גבי מספר מימדי QoS של תשתית הניהול. מימד מפתח ב QoS כולל, אך לא מוגבל, לזמינות, אבטחה, וביצועים. אבני הפינה הללו מבוטאות במושגים אשר ניתנים למדידה ב SLAs. ניתן לתאר יכולות אבטחה כזכויות המסופקות על ידי מחלקה או שירות המיוחס למשתמש. מאפייני הביצועים יכולים להיות מוגדרים במושגים ומטרות של זמני תגובה או זמני עיבוד ידועים-היטב.

8.5.1.ביצועים (Performance)

מספק מדידות ודוחות- מידע כיצד מערכת מתפקדת, כגון עומס CPU מצטבר, אשר מתבטא בביצועי מחשוב של שירות ספציפי

8.5.2.זמינות (Availability)

באופן פרטני נציין את המטריצה אשר מראה את תכיפות הכשלים של שירות – מכל הסוגים, כולל המקרים במאקרו כגון קריסת מערכת ו\או מקרים פחות ברורים, כגון כשלון לספק את השירות כתוצאה מנפילה ביכולת התפוקה של השירות.

8.5.3.אבטחה (Security)

האבטחה חשובה מאוד בנקודות ובאספקטים של סביבת ה IT. מדדים אלו כוללים הפרות אבטחה, סבירות בהפרות אבטחה, וניהול זהויות כגון הזמן להגדיר או למחוק משתמש וכדומה.

8.6. אינטראקציות עם שאר רכיבי OGSA

ניהול-עצמי מבצע אינטראקציות כמעט עם כל שאר האספקטים של OGSA. להלן מספר דוגמאות כדוגמה.

- **איתור** למציאה וקישור משאבים חדשים.
- **ניטור** ומעקב במטרה לספק את המידע הנדרש לקבוע את מצב המערכת.
- **הזמנת משאבים** במתרה לנצל ולחזות את ניצולת המשאבים.
- **תהליכי זרימת עבודה [workflow]** לביצוע אוטומציה של פעולות אשר ה SLMs צריכים להתבצע כאשר מתקיימים תנאים של פעילות חריגה.
- **קומפוזיציה** אשר מטרתה ליצר רמה גבוהה יותר של SLMs מכאלו פשוטים יותר.
- **אבטחה** ובאופן פרטני הרשאות והזדהות, חיוניים ברכיבי מערכות שונות אשר יכולות מעורבות בפעילויות הניהול.
- **ניהול משאבים** ובאופן פרטני מודל ניהול שירות או משאב, נדרש לספק את הייצוג של שירות או משאב אשר הוא מנוהל. ה SLMs יכולים לקרוא את המידע, ויכולים לתפקד על-פי החוכמה שבו, כתשובה לסביבה משתנית או פקודה מרמה גבוהה יותר.

9. שירותי מידע (Information Services)

9.1. מטרות (Objectives)

היכולת לגשת ולשנות בצורה יעילה מידע לגבי אפליקציות, משאבים, ושירותים בGrid היא יכולת חשובה של ארכיטקטורת OGSA. בפרק זה, המונח מידע מתייחס למידע דינאמי או אירועים אשר משמשים לניתור הסטאטוס של המשאבים בGrid.

בפועל, שירותי מידע צריכים לתמוך במגוון דרישות QoS לשם אמינות, אבטחת מידע, וביצועים. למרות שניתן לתכנן שירות אשר מתמודד עם כל דפוסי המימוש ו QoS, הכי טוב לשרת את התפישה של OGSA על ידי מספר שירותים, חלק מהם הנם שירותים כלליים, וחלק אופטימאליים להגשמת מקרה שימוש (Use Case) ספציפי.

למרות זאת, צריך להיזהר לא לסיים עם שירותים לא שלמים. האתגר הוא לאזן את בין ה –"**כלליות**" והדרישות הנובעות מסמנטיקה ו QoS בתחום ספציפי. בתיאוריה, "**כלליות**" יכולה להשיג הפשטה של יכולות נפוצות ופונקציונאליות אשר מכסות את הדרישות הנפוצות ברוב מקרי השימוש ככל שניתן. כאשר מחד הסמנטיקה המשותפת יכולה לעודד קישוריות, מאידך היא יכולה להיות מופשטת מידי ולא לחשוף בצורה מלאה את ההתנהגויות הרצויות.

לקוחות של שירותי המידע של OGSA כוללים, אך לא מוגבלים לשירותי ניהול ההפעלה, שירותי חשבונאות, שירותי זיהוי תקלות, שירותי הזמנת משאבים, שירותי ניצולת משאבים, וניטור אפליקציות. בכדי לפשר את הקישוריות והשימוש החוזר, שירותי המידע עצמם צריכים להבנות על גבי תשתית היכולות של OGSA כגון התראות (WS-Notification). שירותי המידע יכולים להשתמש ביכולות OGSA כגון גישה לעיבוד שאילתא מבוזרת.

9.2. מודלים (Models)

האפיון של שירותי המידע תלוי באופן נרחב על עובדות כגון הדרישות שמוצבות על מקור המידע, ודרישות הבטחת רמת השירות. בכל אופן אנו רואים תבניות חוזרות בשירותי המידע. צרכנים וספקים של מידע צריכים להיות בצימוד רופף ולא להיות בעלי ידע מקדים אחד על השני. הצרכנים יכולים להתקשר עם הספק או מתווך ולמשוך ממנו מידע כך שבקריאה אחת הם יכולים להשתמש במנגנוני מנויים ולקבל מידע כאשר המידע הופך להיות זמין.

ארכיטקטורת OGSA לא מתארת מראש את מודל המידע שצריך בכדי לממש שירות מידע או את השפה לאיתור המידע. מערכות נוכחיות לרוב נופלות לתוך אלו המבוססות על שפות כגון XML ו XPath/XQuery או במודל יחסי ושאילתות SQL. מטא-מידע מקושר למידע לשם תיאור המבנה של המידע והשימוש בו. לצורך קישוריות, נדרשת הגדרה של סכמת אירועים בעבור שירותי המידע. במקרים מסוימים, כאשר ביצועים הם לא בעדיפות עליונה והקישוריות נפתרה, אירועים אופטימאליים על-פי הגדרת-משתמש יהיו יותר מתאימים. שירותי מידע יכולים לדוגמה לאפשר לספקים ולצרכנים לגלות אחד את השני על ידי כך שהם מספקים מידע מפורט לגבי עצמם לשם חיפוש ושאילתא. משרד

רישום מיוחד [registry] או מנגנון נקודה-לנקודה יכול לשמש לצורך מטרה זו. התיאור יכול לכלול את סוג המשתמש או הצרכן, סוג המידע שהם מספקים או צורכים, ואת נקודות האינטגרציה איתם והכתובות שלהם [URL].

9.3. תסריטי דוגמה (Example scenarios)

9.3.1.תסריט ספריה (Directory scenario)

משתמש מעוניין לאתר תיאור של שירות אשר תואם לקריטריונים ופונקציונאליות נדרשת בעבורו. הוא מבצע שאילתא בספריה מרכזית בה מופצים כל התיאורים של כל השירותים, ומקבל תשובה. בעל הספרייה מחליט מי יכול לפרסם מידע לספריה ומי מורשה לחפש בה. שירות UDDI הוא דוגמה של שירות מורשת (Legacy) מסוג ספריה. שירות הספרייה של OGSA יכול להיות מבוסס על רעיונות הקיבוץ של WSRF (WSRF Group Concepts).

9.3.2.תסריט המעקב (Logging scenario)

מספר של תסריטי מחשוב בסביבה מבוזרת דורשים יכולות מעקב. תסריטים אלו כוללים את היכולת להבין ולהחליט על תקלות, מדידת השימוש, התאוששות מתקלות, עיבוד טרנזקציות, ואבטחה. איתור התקלות יכול להתבצע כך :

המפתח כותב קוד המטפל בהתאוששות מתקלה באפליקציה. מהפתח עוקב אחרי הנחיות הארגון למעקב והוא מכנים קריאות כתיבה ללוג בקוד שהוא מספק השר מתעד כל התנהגות חריגה. הוא אחראי לקידוד החלק באפליקציה אשר ניגש ל RDBMS להשיג ולאתר את המאפיינים של משאב חלופי. הוא מקודד את החלק באפליקציה שכאשר יש כשלים ב RDBMS הם נתפסים ונכתבים רשומות בלוג. לאחר שהפיתוח הסתיים, האפליקציה מותקנת לסביבה תפעולית בה הרשומות מעובדות על ידי שרשרת רכיבים שמתמחים לטפל בבעיה.

בסביבה שכזו, אשר מבוססת על רמת החומרה, רשומות נאספות בקובץ על-פי רמת הגדרת הגרת המתפעל. בשל ההתקנה של שירות מדיניות חדש, הקוד הגיע למצב חריג ונרשמת ללוג רשומה המצביע שתשתית ה RDBMS דחה את הגישה לאפליקציה המתפעל הגדיר את שסביבת התפעול לשמור רשומות ללוג ברמה זו.

9.3.3.דפוס צרכן \ ספק (Producer/Consumer patterns)

לרוב, צרכנים וספקים יכולים לתקשר באופן ישיר אחד עם השני. במקרים בהם גישה ישירה בין הצרכן לספק אינה מתאימה או לא מתאפשרת, הדפוס הבסיסי הוא צימוד רופף בין הספק לצרכן בעזרת מתווך (Intermediary) גישה זו מאוד נפוצה. בתבנית **ספק-מתווך-צרכן**, ספקים שמים מידע במתווך, וצרכנים שולפים את המידע ממנו, במובן הכללי, דפוס זה נכון לכל מאגר נתונים. כלומר ספקים כותבים את במידע לקובץ ,ODBMS, RDBMS וכו׳.

בנוסף לתמיכה בממשקי הספק וצרכן, מתווך צריך לתמוך בממשק ניהול. ממשק הניהול שולט באותן פונקציות אשר לא מיוחסות באופן ישיר אל פעולות הקריאה והכתיבה למאגר הנתונים. פעולות

השולטות במדיניות וכדומה. בסביבה מחשוב מבוזרת, לדפוס המדובר נוסף מספר שירותי תשתית שתפקידם לספק מידע לצרכנים. אנו מתייחסים לשירותים אלו כשירותי מידע. הם כוללים שירות שמות, שירותי מעקב, ושירותים נוספים.

אנו מניחים שהקורא מכיר את תפיסות אלו. כל אחד מהשירותים הללו התפתח כדי לתמוך בסדרה ברורה של סמנטיקה ואיכויות של שירות. ה - WS-Notification מספק יכולות ליבה של ממשקים אשר ינוצלו על ידי שירותי המידע של OGSA.

המאפיינים שלו כוללים מודל pub/sub, מנגנון לארגון פריטים לפי עניין המנוי הידוע כ "Topic", וממשקי ניהול לשם פרסום ומנוי. שאר הגדרות שירותי הרשת שמתייחסים להבטחת רמת השירות המסופקת על ידי תשתית המסרים צריכה לקבל התייחסות לדוגמה [WS-Reliability] המאפשרת העברת מסרים אמינה לנוכח כישלונות בתשתית התקשורת.

9.3.4.תסריט ניטור ובקרה (נו"ב)

משתמש מעוניין לאתר את אותם משאבים אשר דורשים תחזוקה. לדוגמה תורים, דיסקים, וכדומה אשר התפוסה שלהם מגיעה ל 90-80 אחוז המשתמש מבצע שאילתא התואמת לקריטריונים ופונקציונאליות נדרשת בעבורו. הוא מבצע שאילתא בספריה מרכזית בה מופצים כל התיאורים והמאפיינים של כל המשאבים, ומקבל תשובה. מנהל הספרייה מחליט מי יכול לפרסם מידע לספריה ומי מורשה לחפש בה על-פי מדיניות אבטחת המידע.

9.4. יכולות פונקציונאליות (Functional capabilities)

במסגרת שירותי המידע בארכיטקטורת OGSA אנו מגדירים את היכולות הבאות: מתן-שמות, איתור, העברת מסרים, מעקב, ויכולות ניטור.

9.4.1.סכמת שמות (Naming scheme)

מערכות מסורתיות מבוזרות לרוב תומכות בשתים או שלוש שכבות של סכמת שמות. ב OGSA שירות מתן השמות,ה OGSA-naming, משתמש בשלוש רמות של מוסכמות. כל ישות בעלת שם מקושרת עם שם (אופציונאלי) אנושי ידידותי, שם מופשט, וכתובת. השם האנושי הוא לרוב שם שאדם מסוגל לקרוא ויכול להיות מקושר לטווח שמות. טווחי שמות בדרך כלל היראכיים ובעלי הגבלות סמנטיקה. טווח שמות היראכי מאפשר לכל שם להיות ממופה לנושא מסוים. לא חובה שהשם האנושי יהיה חד ערכי. השם המופשט הוא שם עקבי אשר לא מוגדר במקום מסוים. שאר המאפיינים של שם מופשט כגון ייחודיות עדיין בבחינה. נקודות קצה של כתובת WSRF בעלת renewable reference הם דוגמה לשם מופשט, הכתובת שמגדירה את מיקום הישות ברשת. דוגמה לכתובת היא הקומבינציה של כתובת ומאפיינים בנקודת קצה WSRF, כתובת זיכרון, כתובת IP. ב OGSA אנו מניחים את הקיום של מצביע למשאב. מצביע למשאב הוא שם מופשט של המשאב המקושר אליו (אם יש).

9.4.2.איתור (Discovery)

יכולת אוניברסאלית נדרשת היא היכולת לאתר שירות או משאב. ספרייה (או registry) הוא פתרון ברור, אבל לא היחיד. ספריה מופרדת משאר הפתרונות האפשריים בכך שהיא בעלת מאגר של המידע האחרון והוא אופטימאלי לחיפושים. נדרשים זמני תגובה קצרים ועוצמות חיפוש גבוהות. הספרייה יכולה להיות מועתקת לצרוך יכולת גידול. חלופה, יכולה להיות index כמו goggle שלא בדומה ל registry, אשר נוטה להיות בשליטה מרכזית, כל אחד יכול ליצור index.

אופציה נוספת היא איתור ושאילתות peer-to-peer , שבו שירות רשת הוא נקודת קצה ברשת של משקיפים ובאופן דינאמי מתשאל את השכנים שלו במטרה למצוא התאמה. השאילתא רצה ברשת דרך נקודות הקצה מאחד לשני עד אשר נמצאת התאמה, או מספר קפיצות מקסימאלי התבצעו, או קריטריון ביטול אחר התקיים. עדיין קיימת אלטרנטיבה נוספת לאיתור שירות המבוססת על שירות GMA (ראה למטה).

9.4.3.העברת מסרים (Message delivery)

צרכנים וספקים מתקשרים על ידי החלפת מסרים, וזה יכול להתבצע על ידי תשתית מסרים נפוצה. תשתית זו דואגת אך ורק להפצת המסרים לשותפים המעוניינים, ולא כיצד המסרים האלו נבנו במקור.

ספקים שולחים מסרים ישירות לצרכנים הרלוונטיים או נעזרים במתווך (message broker) אשר מפריד בין הצרכן לספק. במקרה מאוחר יותר הספקים מפיצים את המסרים שלהם למתווך, אשר לוקח אחריות להעברת המסרים לשותפים שמעוניינים בהם.

ספק (או מתווך) יכול לספק יכולות התראות ופונקציונאליות נוספת כגון חיפוש שירות. מתווכים יכולים לשמור הודעות ולהעביר אותן דרך מאגר יציב לשם העברה אמינה של המסרים.

9.4.4.תיעוד (Logging)

שירות התיעוד ב OGSA מתפקד כמתווך לאלמנטים של לוג בין הצרכנים לספקים. ספקים כותבים לוג, וצרכנים קוראים אותו (אבל לא מעדכנים). בכדי להבטיח את הקבלה הכללית של סמנטיקת לוג בסיסית במטרה לאפשר תחקור של מימושים קיימים, שירותי הלוג של OGSA צריכים לתמוך ביכולות נפוצות הקיימות בפתרונות תיעוד. יכול להיות מספר ספקים וצרכנים למתווך תיעוד ספציפי, והם מסוגלים לספק מסננים של רשומות. בשירות התיעוד תקשורת המסרים אופטימאלית לביצועים ורשומות נשמרות לתקופה מסוימת.

9.4.5.ניטור (Monitoring)

שירות ניטור של OGSA יכול לשמש הן לאפליקציות והן למשאבים באופן שווה. ביישומים בעלי אופי מיוחד (לדוגמא – יישומי RT), ייתכן שידרשו דרישות מיוחדות משירות הניטור.

9.4.6.מידע כללי ושירותי ניטור (General Information and Monitoring service)

באופן כללי שירות OGSA אשר מספק קומבינציה של היכולות המתוארות למעלה יכול לספק גמישות בעבור משתמש הקצה. למשאבי Grid באופן כללי (כולל אפליקציות ושירותים), כמות המידע הזמין לגבי המשבים יכול להיות גדול ומפוזר על גבי הרשת ולהיות מעודכן באופן תדיר.

חיפושים במרחב הזה יכולים להיות בעלי עיקובים לא מקובלים. במטרה לנהל את מידע זה בצורה נשלטת, חשוב להפריד בין מגנוני גילוי מקורות מידע למנגנוני שינוע מידע. מנועי חיפוש אמורים רק לאתר מקורות מידע. צריכה להיות קיימת ספריה בעלת תפקיד מיוחד לשם אחזקת מטא-מידע לגבי המשאבים. במקרה כזה ספרייה צריכה להתמודד עם תדירויות גבוהות של עדכונים אשר מצופים לבוא מסביבה דינאמית כגון OGSA.

צמד ספק\צרכן אינדיבידואלי יכול להגביל את כמות המידע שזורם ביניהם בצורה שתספק את הדרישות של שאילתת הצרכן. מודל זה שונה מברוקר אשר משלב את המנגנון של מציאת המשאב ומפיץ את המידע ומבצע את העברה לתוך ערוץ אחד. משתמש של שירות כללי צריך להיות מסוגל לשים מידע, מבלי להתייחס למטרת השימוש בשירות, ומבלי להצטרך להבין את מורכבות המערכת. דבר ראשון צריך להגדיר איזה מידע הופך להיות זמין ב OGSA, והיכן שהסוג ומבנה המידע צריך להיות מוגדר היטב. המשתמש עלול לרצות להגדיר מאפיינים ומדיניות. זה כולל כיצד מידע מוחזק, הבטחת אספקה, ושליטה בגישה. הצרכן יכול לסנן מידע לפי נושא. משתמשים מתקדמים יותר עלולים להידרש בהבנה יותר מעמיקה בנושאי הקרביים של השירות.

9.5. מאפיינים (Properties)

9.5.1.אבטחה (Security)

חוקי הרשאות והזדהות לצרכנים וספקים מאפשרים להם להחליף מידע בצורה מאובטחת. איתור מטא-מידע לגבי ספקים וצרכנים יכול להיות נתון לחוקי אבטחת מידע. חלק מהשירותים זקוקים שהמסרים יהיו מאובטחים לשם העברתם.

9.5.2.הבטחת איכות השירות (Quality of service)

רמות שונות של הבטחת איכות יכולות להיות מסופקות על ידי שירותי המידע של OGSA, כגון שליחה אמינה באמצעותWS-Reliability, כפי שמוגדרת על ידי תקן של OASIS.

9.5.3.זמינות ביצועים ויכולת גידול (Availability/Performance/Scalability)

שירותי מידע משחקים תפקיד קריטי ב OGSA. מכיוון שכמעט כל שאר היכולות ב OGSA עושים שימוש בהם, הם צריכים להיות זמינים כל הזמן ולהיות סובלניים לכישלונות חלקיים. הרבה לקוחות של שירותי המידע צפויים לקבל מידע בתדירויות גבוהות ולא יכולים להרשות לעצמם לחכות תקופות זמן ארוכות. במקרה הזה נדרשות מערכות בעלות יכולות ביצועים גבוהות . וזאת מכיוון שמספר רב של משאבים ושירותים, מעוניים ליצר ולצרוך מידע, המערכת צריכה להיות בעלת יכולת גידול על גבי שטחים רחבים ורשתות.

9.6. אינטראקציות עם שאר רכיבי OGSA (Interactions with the rest of OGSA)

השימוש במודלים ופרוטוקולים סטנדרטיים מקלים את העברת המידע בין ספקים לצרכנים. השימוש באירועים סטנדרטיים מוכרים מאפשר את הגילוי והעיבוד של אירועים ממגוון רחב של משאבים פוטנציאליים ברחבי ה VO.

באופן אידיאלי, כדי להימנע מעלויות תיווך (Mediation), מודל האירועים במשאבים (Resource) ומודל האירועים בתשתיות (Infrastructure) צריכים להיות בהלימה.

כמו כן, על מנת להתאים לאירועים בקבוצה ספציפית, כמות האירועים צריכה להיות ניתנת להרחבה.

סכמת האירועים כרגע נמצאת בפיתוח על ידי OASIS WSDM-TC.

מנגנוני ההתראות, או ההרחבה שלהם יכול לעזור בהעברת האירועים להספקים לצרכנים ברגע שם גילו אחד את השני, ובין שאר הרכיבים של המערכת.

שירותי אבטחת מידע נדרשים לשם ביצוע הזדהות והרשאות בין הרכיבים השונים.

יכולת לעיבוד שאילתות מבוזרות נדרשות לביצוע שלב תכנון השאילתא.

מנגנוני העתקה נדרשים, כך שמאגרי מידע-העל יועתקו במטרה להימנע מנקודת כשל אחת.

נספח ה – דוגמא לפעילות מחקרית בנושא ה-SOA וה-Grid[3]

תכנית CCRP (C4I Cooperative Research Program) הינה תכנית מחקרית המתנהלת במשרד ההגנה האמריקאי באה לשפר את ההבנה של השלכות עידן המידע על הביטחון הלאומי. במסגרת התכנית מוגדרת תפיסה של לוחמה ממוקדת מידע המדגישה את חשיבות הרשת והקישוריות (Network Centric Warfare) ולא ממוקדת מערכות (System Centric Warfare), כאשר הרעיון המוביל למימוש הוא תפיסת ה Global Information Grid (להלן GIG) המושתת על בסיס פרדיגמת רשת השירותים (SOA).

ה- GIG הנו רשת גלובלית מבוססת Grid המחברת מידע, תהליכים ואנשים על מנת לאסוף לעבד, לשמור, לנהל ולהפיץ את המידע לטובת הגורמים המבצעיים, תומכי הלחימה, מקבלי החלטות וכדומה.

החזון של ה- GIG הוא מעבר מהותי בדרך שמידע מנוהל מתוקשר ומאומת. מערכת ה GIG תספק למשתמשים סביבת קישוריות מידע אשר עונה על צורכי Real-Time ו Near-Real-Time של לוחמים ומשתמשים עסקיים. ה GIG משתמש בטכנולוגיות מסחריות אשר הורחבו כדי לתת מענה לצורכי משרד ההגנה האמריקאי לצורכי משימות קריטיות.

מטרת ה- Global Information Grid לשמש כמערכת מוכוונת רשת הפועלת בהקשר גלובלי ומספקת עיבוד, אחסון, ניהול, ושינוע של מידע במטרה לתמוך במשימות משרד ההגנה, האבטחה הלאומית וקהילות מודיעין, זרועות מבצעיות אסטרטגיות טקטיות, ועסקיות בזמן לחימה ובעת שלום, תוך חשיפת שירותים ויצירת יכולות לצרוך אותן בצורה דינאמית (Composite Applications).

יכולות ה GIG יהיו נגישות מכל מקום מבצעי בסיסים, עמדות, תחנות, מתקנים, פלטפורמות ניידות, ואתרים נפרשים. ה GIG מתקשר ומקשר עם בעלות ברית, קואליציות, ומערכות שאינן מערכות GIG.

המטרה הסופית של חזון ה GIG הוא לספק לרשות הפיקוד הלאומית, לוחמים, אנשי משרד הביטחון, קהילות המודיעין, משתמשים עסקים, מכתיבי מדיניות ואנשים שאינם אנשי משרד הביטחון מידע, עליונות, עליונות בקבלת החלטות וקשת מלאה של עליונות דומיננטיות.

[3] מבוסס על הפרסומים של משרד ההגנה האמריקאי [22]

In the second stage, scenarios and quality properties were defined to perform quality comparison between the architectures, which are based on the Tradeoff Analysis Model methodology (model of qualitative comparison of architectures). The comparison examines the capabilities of the new architecture, especially in the face of the existing architecture's limitations.

The comparison results expose the benefits of the new architecture in coping with exposure of large-scale services, while maintaining dynamic distributed operation, without compromising on information security and the consumption thereof.

The study, which presents an innovative approach, is part of a new computing era, of which we begin to experience recently under the title of cloud computing services (Cloud Computing). These services are actually a live demonstration of the cross-border grid concept, crossing the internal and external boundaries of the organization, allowing consumption of services in a decentralized and dynamic manner.

The proposed architectural infrastructure, together with its recommendations in the area of virtualization, standardization and sharing, are important elements for further development of the comprehensive concept of cloud computing. The proposed architectural foundation promotes the computing capabilities required by the services consumers and suppliers, in order to provide application services in the cloud era.

Abstract

In the recent years we have witnessed a penetration of internet architectures and technologies into the corporate computing world, such as web service technologies. These technologies and architectures are enabling the development of new work concepts based on dynamic consumption of services, such as Service Oriented Architecture (SOA).

The study introduces the ability to produce new services platforms, based on distributed architectures allowing the consumption of services, which can originate from various sources, including outside the organization in secured and decentralized manner. The study focuses on the innovation of the Grid technologies and its compliance with the requirements of the SOA services paradigm and the Enterprise Service Bus (ESB) infrastructure layer.

The study introduces the ESB concept as a main infrastructure for exposure of services and their consumption in the SOA era, based on a comprehensive research in this field, conducted by the Department of Defense in U.S. Army. The definitions created by the U.S. army adopt the definition of nine families of basic services, which constitute a definition and characterization of the core capabilities required for this distributed infrastructure.

In the first phase, the study presents a new grid-based architecture, designated for the application of services infrastructure platform (nine families of base services), that will enable their exposure and consumption on the web on standard level, without the need for a central mediator and in a completely distributed manner - Grid based ESB. This architecture relies on technological innovation and implementation of Grid services, designated to meet the problems and limitations of the existing architecture. Current ESB solutions based on Web Services, focusing mainly on the decentralization and dynamic issue, which constitutes a barrier for growing in the existing solutions.

Table of content

This work was carried out under the devoted supervision of Professor David Schwartz, Department of Business Management, Bar-Ilan University.

Reference Architecture for a Grid Oriented Enterprise Service Bus

Ronen Yochpaz

Business Management School

Ph.D. thesis

Submitted to the Senate of Bar-Ilan University

Ramat-Gan, Israel July 2010

Reference Architecture for a Grid Oriented Enterprise Service Bus

Ronen Yochpaz

Business Management School

Ph.D. thesis

Submitted to the Senate of Bar-Ilan University

Ramat-Gan, Israel July 2010