# Энтерпрайс Системийн Нэгтгэлд Үйлчилгээнд Суурилсан Архитектурыг Ашиглах Нь

Лхайжав Мөнхдэмбэрэл
Програмчлалын технологийн профессорын баг
ШУТИС-КТМС
Улаанбаатар хот, Монгол улс
lmunkhdemberel@csms.edu.mn

Буянхишиг Мөнхбуян
Мэдээллийн системийн профессорын баг
ШУТИС- КТМС
Улаанбаатар хот, Монгол улс
b.munkhbuyan@csms.edu.mn

***Хураангуй*—Энэхүү судалгааны ажлаар энтерпрайс систем дэх ялгаатай технологиудаар хөгжүүлсэн програм хангамжуудыг хувьд тэдгээрийг бизнесийн шинэ шийдэлд зохицуулахын тулд нэгтгэх боломж, аргачлалуудыг харьцуулан судлав. Ингэхдээ ДотНет болон жава платформ бүхий Oracle болон MSSQL өгөгдлийн сан удирдах системтэй хоёр системийн хувьд өндөр зэрэглэлийн нууцлалтай байх шаардлагын дагуу нэгтгэхдээ үйлчилгээнд суурилсан архитектур ашиглав. Энтерпрайс системийн дэд бүрэлдэхүүнүүд ямар технологи дээр хөгжүүлэгдсэн гэдгээс үл хамааран тухайн системийг дахин зохиохгүйгээр нэгтгэх боломжтой гэдгийг туршиж батлав.**

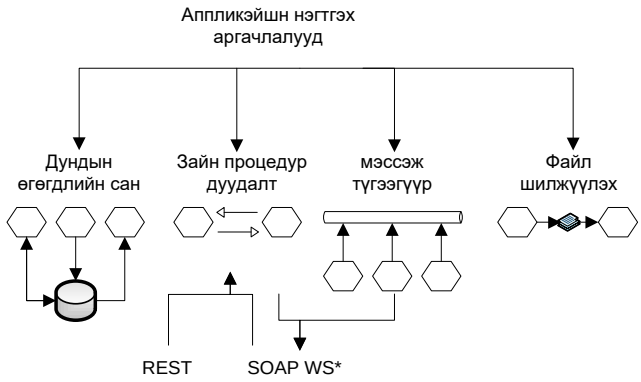
**Түлхүүр үгс —*энтерпрайс систем, үйлчилгээнд суурилсан архитектур, вэб сервис, SharePoint, SOAP, REST.***

## I. Удиртгал

Өнөө үед аливаа байгууллага өөрийн бизнесийн шийдэл, дотоод үйл ажиллагаанд энтерпрайс системийг өргөнөөр ашиглах болсон. Үүнтэй уялдан энтерпрайс систем нь бизнесийн үйл ажиллагаанд зохицсон хурдан хөгжих боломжтой, илүү уян хатан байх шаардлага тулгараад байна [1]. Тухайлбал, шинэ функцийг хуучин системүүдэд нэмэх, хуучин системүүдийг нэгтгэн шинэ функц үүсгэх гэх мэт. Заримдаа үйлчилгээнд суурилсан архитектурыг (YCA) ашиглан их хэмжээний өгөгдлийг шилжүүлэх мөн энтерпрайс системийг эзэмшигч байгууллагууд өөрсдийн шинэ шийдэлд зориулан системд нэмэлт модулийг өргөтгөхийг хүсдэг [2]. Ийм хөгжүүлэлтийн үйл ажиллагаа нь системийг дахин загварчлах, кодчилох зэрэг өртөг зардал, цаг хугацаа, хүн хүч их шаардсан ажил болдог. Харин YCA зарчмыг ашигласнаар тухайн хөгжүүлэлтийн ажиллагааг 50 хувиар бууруулах боломжтой [5]. Мөн тухайн систем нь YCA зарчмын аль стандарттай илүү зохицон ажиллаж байгааг судлан, системийн онцлогт тохирсон стандартыг сонгох нь илүү оновчтой байдаг [1]. Тиймээс оршин байгаа бодит системүүдийн нэгтгэх боломжийг судлах шаардлагатай тулгарч байна.

## II. Аппликэйшн нэгтгэх аргачлалууд

Бие даасан аппликэйшнүүдийг нэгтгэх хэд хэдэн аргачлалууд байдаг. Тухайлбал дундын өгөгдлийн сан, алсын процедур дуудалт, мэссэж (зурвас) түгээгүүр, файл шилжүүлэх (Зураг 1) гэх мэт [8]. Бидний хөгжүүлсэн системүүд нь ДотНэт (Шэйрпоинт) болон Жава флатфортой, Oracle болон MSSQL өгөгдлийн сан удирдах системүүдтэй тул эдгээр аргачлалуудаас мэссэж түгээгүүрийн аргачлал хамгийн тохиромжтой гэж үзэж байна. Мэссэж түгээх үйлчилгээ авахад вэб сервисүүдийг өргөнөөр ашигладаг. Вэб сервисүүд нь өгөгдлийг тархмал байдалтайгаар ашиглах боломжийг олгодог. Мөн програм хангамжийн ялгаатай бүрэлдэхүүнүүдийг хамтарч ажиллах боломж олгоно. Өөрөөр хэлбэл ялгаатай програмчлалын хэл дээр бичигдсэн, өөр компьютерүүд дээр суурилуулсан бүрэлдэхүүнүүдийн хувьд хамтарч ажиллана гэсэг үг юм. Вэб сервисийн өргөн ашиглагддаг үндсэн хоёр төрөл нь SOAP (Simple Object Access Protocol) болон REST (Representational State Transfer) юм [7]. SOAP нь объект хандлагат аргачлал бөгөөд стандарт протоколуудыг ашиглан XML мэссэжүүдийг дамжуулдаг. Програмчлалын хэл болон платформ, шилжүүлэлтийн протоколуудаас (HTTP, FTP, TCP, UDP г.мэт) хамааралгүй байдаг. REST нь нөөц хандлагат аргачлал бөгөөд вэб стандартуудын загварыг ашиглан мэссэж дамжуулдаг. Тухайлбал HTTP болон URI гэх мэт.




Зураг 1. Аппликэйшн нэгтгэх аргачлалууд

## III. SOAP БОЛОН REST

Тухайн стандартуудын гүйцэтгэлийг харьцуулж үзсэн судалгаанд REST аргачлал нь ихэвчлэн илүү гүйцэтгэлтэй байна [6]. Гүйцэтгэл сайтай гэдэг нь нийт түдгэлзэгдсэн хүсэлтийн тоо бага байдаг гэсэн үг юм [7]. Гүйцэтгэл нь REST аргачлалын кэшлэх боломжоос илүүтэй хамаарна. Харин SOAP нь кэшлэлтийг дэмждэггүй.

ХҮСНЭГТ 1. SOAP БОЛОН REST ТҮДГЭЛЗЭГДСЭН ХҮСЭЛТИЙН ХАРЬЦУУЛАЛТ [7]

| **Илгээсэн хүсэлтийн тоо** | **SOAP түдгэлзэгдсэн хүсэлтийн тоо** | **REST түдгэлзэгдсэн хүсэлтийн тоо** |
|---|---|---|
| 60 | 10 | 0 |
| 80 | 59 | 4 |
| 100 | 64 | 9 |
| 120 | 84 | 14 |

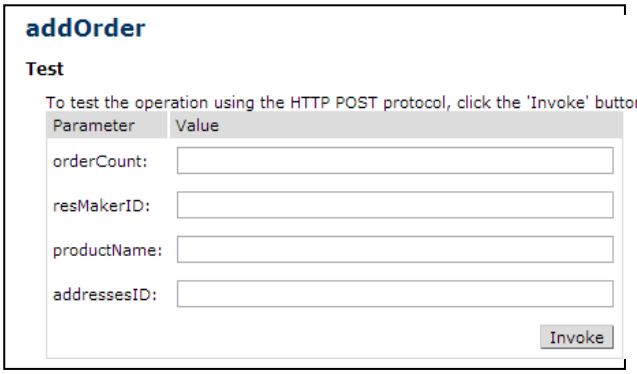
Хүснэгт 1-ээс харахад REST нь илүү найдвартай ажиллах нь харагдаж байна. Мөн REST аргачлалын нэг давуу тал нь олон төрлийн форматыг ашиглах боломжтой байдаг. SOAP нь зөвхөн XML форматыг ашигладаг. Мөн ашиглахад хялбар байдлыг харьцуулан үзвэл REST нь HTTP стандартыг ашигладаг учраас илүү хялбар байна. Аюулгүй байдлын хувьд аль аль нь SSL дэмждэг. Харин SOAP аргачлалын хувьд вэб сервисийн аюулгүй байдлыг мөн дэмжин ажилладаг байна. Өөр нэг давуу тал нь хэрэв ACID (Atomicity Consistency Isolation Durability) транзакшн үйлдэл хийх болсон тохиолдолд зөвхөн SOAP тухайн үйлдлийг зөвшөөрдөг. REST нь зөвхөн энгийн транзакшн үйлдлийг ашиглах боломжийг олгодог тул аюулгүй байдлын хувьд сул талтай байна. Тухайн хоёр аргачлал нь өөр өөрсдийн гэсэн давуу талуудыг агуулж байна. Хэрэв илүү найдвартай ажиллагаа бүхий системийн нэгтгэл болон систем хийж гүйцэтгэх шаардлагатай бол SOAP технологийг ашиглах нь илүү давуу талтай байна. Сонгож авсан системүүдийн хувьд гүйцэтгэлээс гадна аюулгүй байдал нэн тэргүүнд тавигдах тул SOAP аргачлалыг системийн нэгтгэлд илүү найдвартай гэж үзэн нарийвчлав.

## IV. БОДИТ СИСТЕМД НЭВРҮҮЛЭХ

Бидний хөгжүүлсэн системүүд бол жава ADF фрэймворк дээр хөгжүүлэгдсэн вэб аппликэйшн програм болон шэйрпойнт дээр хөгжүүлэгдсэн вэб юм. Харин жава хэл нь SOAP аргачлалын дагуу XML форматтай харьцдаг бэлэн сан JDK – д байдаггүй. Гэхдээ жава хэл дээрх системийн хувьд KSOAP хэмээх jar файлыг ашиглах нь илүү хялбар байна. Шэйрфойнт системийн хувьд ASP.net дээр суурилсан тул вэб сервис буюу asmx бичих боломжтой. Өөрөөр хэлбэл шэйрпойнтын хувьд вэб сервисүүдийг санал болгох юм. Харин жава систем тухайн вэб сервисийг KSOAP ийн тусламжтай хүсэлт явуулан гарсан XML ийг тайлж унших замаар хялбар зохицуулж болно. Судалгааны объект болгон KSOAP 2.6 хэмээх классуудын санг судлав. KSOAP 2.6 – ийн хувьд үйл ажиллагааны үүргийн хувьд гурван төрөлд хуваагдсан 28 классаас тогтсон. Үүргийн хувьд дараах гурван төрөлд хуваагдаж байна.

1. Өгөгдлийг шилжүүлэх;
2. Өгөгдлийг цувих буюу сериалчлах (гол классууд);
3. Алдаа болон форматыг хариуцах;

Энэхүү 28 классын тусламжтай XML өгөгдлийг сүлжээгээр дамжуулах, тайлж унших зэрэг асуудал нь илүү хялбар болж байна. Харин шэйрпойнт системийн талаас вэб сервис бичигдэнэ. Тухайн вэб сервис нь параметр авч боловсруулах байдлаар зохиомж гаргасан. Мөн жава системийн хүсэлтийн дагуу WSDL (Web Service Description Language) буюу вэб сервис тодорхойлолтын хэлний дагуу XML хариуг буцаана. Тухайн буцаж ирж буй үр дүнг жава системд тайлан унших замаар хоёр системийг нэгдсэн байдлаар илүү үр ашигтай ажиллах боломж олдох юм. Жишээ болгон энгийн вэб сервис бичсэн бөгөөд тухайн вэб сервис XML формат буцааж байв. Дөрвөн параметр аваад үр дүнд нь өгөгдлийн сан руу параметрүүдийг нэмдэг захиалга нэмэх (addOrder) гэсэн цонх бүхий програмыг ДотНэт програмчлалын хэрэгсэл ашиглан боловсруулж туршив (Зураг 2). Үр дүнд нь:




Зураг 2. Вэб сервисийн үр дүн

Дараах кодын хэсэг нь бичсэн вэб сервисийн үр дүн бөгөөд параметрүүдийг дамжуулан туршиж үзвэл XML формат бүхий хариу үзүүлж байв. Харгалзах XML кодыг үзүүлбэл:

```
<s:element name="addOrder">
<s:complexType>
<s:sequence>
<s:element minOccurs="1" maxOccurs="1"
     name="orderCount" type="s:int"/>
<s:element minOccurs="1" maxOccurs="1"
     name="resMakerID" type="s:int"/>
<s:element minOccurs="0" maxOccurs="1"
     name="productName" type="s:string"/>
<s:element minOccurs="1" maxOccurs="1"
     name="addressesID" type="s:int"/>
</s:sequence>
</s:complexType>
</s:element>
```

Вэб сервис XML хариу үзүүлж байгаа тул жава систем дээрээс тухайн XML – ийг хүлээн аваад тайлахад л хангалттай. XML - ийг хүлээн авч тайлах програмын код нь жава хэл дээр дараах байдлаар бичигдэнэ. Үүнд:

```
private static String SOAP_ACTION =
        "http://tempuri.org/addOrder";
private static String METHOD_NAME = "addOrder";
private static String NAMESPACE =
"http://tempuri.org/";
private static String URL =

"http://10.0.2.2:56984/Service1.asmx?WSDL";
```

Дээрх классад вэб сервисийн нэр болон URL хаягийг гишүүн болгон тодорхойлж өгсөн. Туршилтыг нэг компьютер дээр гүйцэтгэсэн тул вэб сервисийн хаяг нь дотоод IP хаяг байна.

```
SoapObject Request = new
SoapObject(NAMESPACE,
       METHOD_NAME);
Request.addProperty("orderCount",
OrderCount);
Request.addProperty("resMakerID",
resMakerID);
Request.addProperty("productName",
       productName);
Request.addProperty("addressesID",
addressesID);
```

Шэйрпойнт систем рүү параметр дамжуулахдаа `Request.addProperty("orderCount", OrderCount);` гэсэн байдлаар дамжуулдаг. Энэ нь orderCount параметр рүү OrderCount жава хувьсагчийг дамжуулж өгөх кодын хэсэг юм. Харин дараах хэсгүүдэд ДотНэт вэб сервис рүү хүсэлт явуулахыг дараах байдлаар зарлаж бичсэн. Үүнд:

```
SoapSerializationEnvelope soapEnvelope =
new
SoapSerializationEnvelope(SoapEnvelope.VER1
1);
soapEnvelope.dotNet = true;
soapEnvelope.setOutputSoapObject(Request);
```

Http ашиглан мэдээлэл шилжүүлэх хүсэлтийг зарласан кодын хэсэг дараах байдалтай байна. Үүнд:

```
HttpTransportSE HttpTransport = new
       HttpTransportSE(URL,60000);
HttpTransport.call(SOAP_ACTION,
soapEnvelope);
return result.toString();
```

Туршилтын үр дүнд тухайн хоёр бие даасан системүүд хоорондоо маш сайн хамтарч ажиллаж байв. Жава болон Дот.Нэт суурьтай ялгаатай системүүд бүхий энтерпрайс системийн хувьд нэгтгэх асуудал нь энэ аргаар шийдэгдэх боломжтой нь батлагдлаа.

## V. ДҮГНЭЛТ

Энтерпрайс системийн нэгтгэлд тухайн хоёр аргачлалыг ашиглаж болох бөгөөд аргачлал тус бүр өөрсдийн сул болон давуу талтай байв. Нэгтгэлд ашиглаж болох дөрвөн төрлийн арга байснаас бидний системүүдийн тохиолдолоос хамааруулан хоёр аргыг нарийвчлан харьцуулж судлав. Харин системийн шаардлагын дагуу аюулгүй байдлыг нэн тэргүүнд тавих шаардлагатай байсан тул SOAP аргачлалыг илүү нарийвчлан авч үзлээ. Тухайн аргачлалын дагуу XML формат бүхий WSDL болон жижиг хэмжээний вэб сервисийг үүсгэн, хоёр системийг хялбар нэгтгэж болох нь харагдлаа. Өөрөөр хэлбэл энтерпрайс системийн дэд бүрэлдэхүүнүүд ямар технологи дээр хөгжигдсөн гэдгээс үл хамааран тухайн системийг ахин бичихгүйгээр хялбар нэгтгэх боломжтой гэж дүгнэв.

## АШИГЛАСАН МАТЕРИАЛ

[1] Zaigham Mahmood. “Service Oriented Architecture: A New Paradigm for Enterprise Application Integration”. Proceedings of the 11th WSEAS International conference on computers, Agios Nikolaos. July 26-28 2007.

[2] Antynio Martins, Pedro Carrilho, Miguel Mira da Silva and Carlos Alves. “Using SOA paradigm to Integrate with ERP Systems”.

Department of Computer Science, Instituto Superior Técnico, July 26-28 2007.

[3] Martin Huvar, Timm Falter, Thomas Fiedler, Alexander Zubev. Developing Applications with Enterprise SOA. SAP Press July-31-2008. pp 1-34.

[4] Thomas Erl. SOA Design patterns. January-9-2009. pp 464-515.

[5] WestWrax Applications. SAP ROI (2004)

[6] Ricardo Van Den Broek. “Comparing performance of SOAP and REST PHP clients”. University of Twente

[7] Feda AlShahwan, Klaus Moessner, Francois Carrez. “Evaluation of Distributed SOAP and RESTful Mobile Web Services”. International journal on Advances in Networks and Services vol 3. Year 2010.

[8] Cesare Pautasso, Olaf Zimmermann, Frank Leymann. “RESTful web services vs. ‘Big’ Web Services: Making Right Architectural Decision”. April 21-25, 2008.