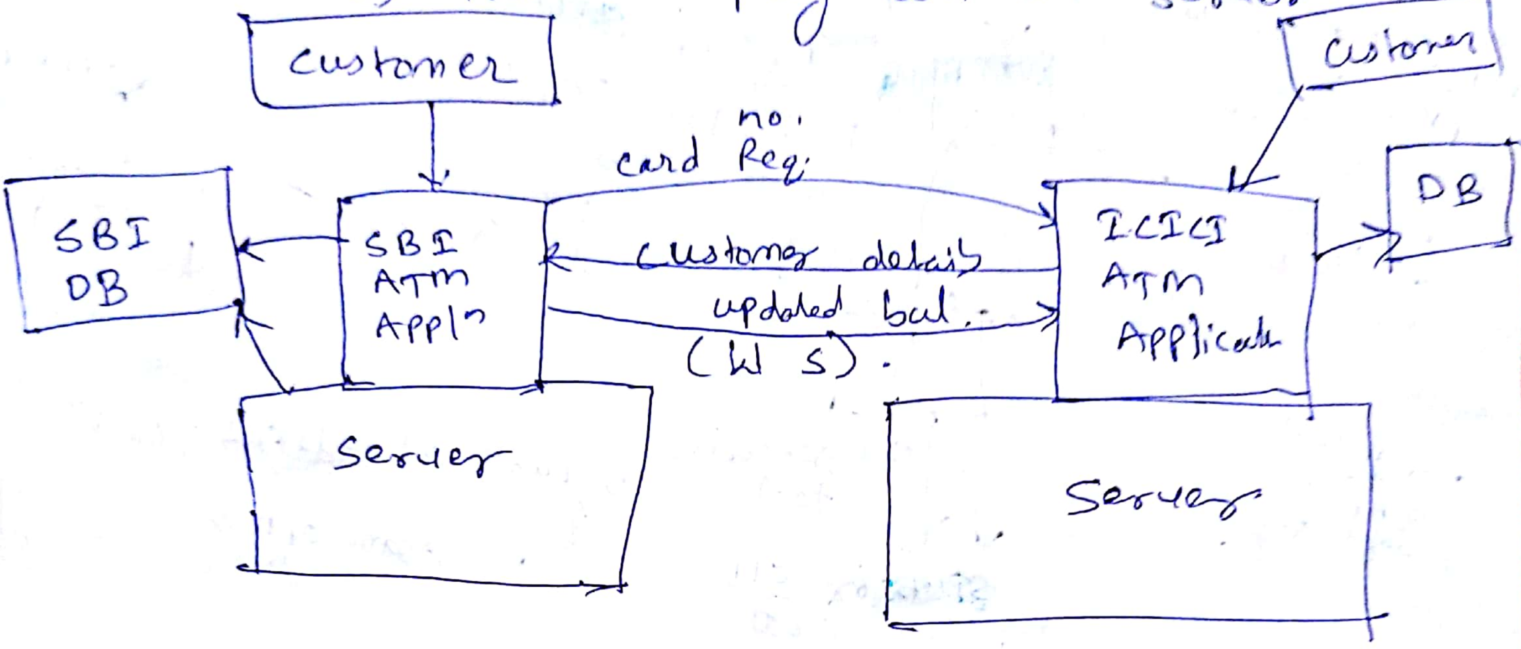
* Web Service

1) SOAP
2) REST

↳ Core Java
↳ Java web application
↳ How to deploy WA on server.



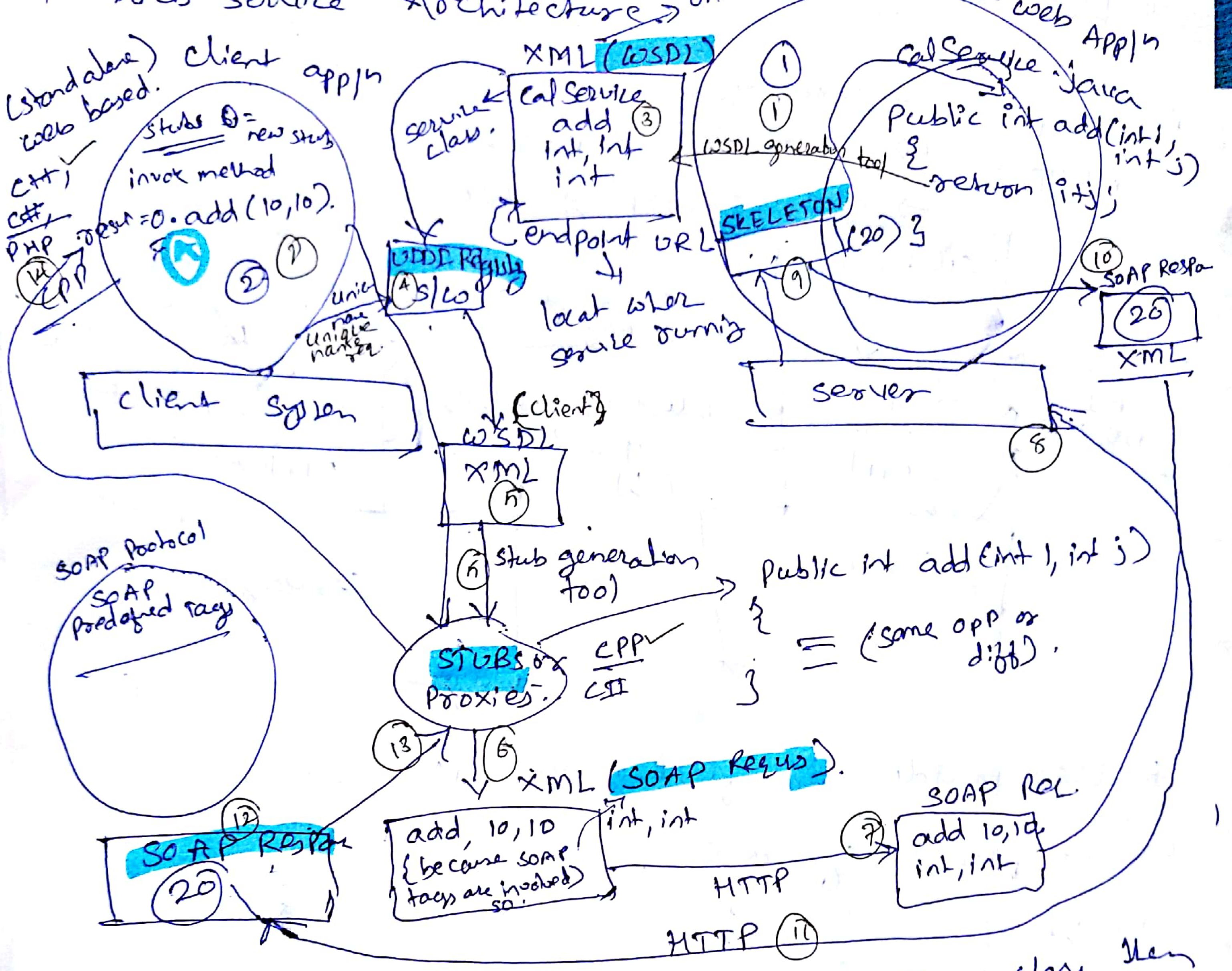
* WS provides specification not API

↳ Components of WS

1) WSDL
2) UDDI → optional.
3) SKELETON
4) STUB
5) SOAP protocal.
6) HTTP protocal

Scanned by CamScanner

* Web service Architecture



→ SKELETON - is Predefined class, if service is Java class then skeleton is Java class (use Predefined tags from SOAP tg)

→ to prepare { to know the client about your service }

→ WSDL - its an XML file which contains service class details, ~~who~~ service provider will prepar WSDL (to prepar we required WSDL tool (i/p → service class, end point URL).

→ UDDI. Registry : store WSDL file in UDDI Registry with unique name (UDDI Registry is not mandatory).

↳ we can send WSDL file through email also.

* client will create STUB object & by using Stub object he will invoc method by passing paramets

↳ STUBS | proxies : To prepare Stubs we required STUBS Generator tool

- As if client appln is in CPP then STUBS should be In CPP
- As if cliet appln is in PHP then STUBS shoul be in PHP
- As if cliet appln is in Java then STUBS sho be in Jar

* SOAP - It is a messaging protocol
* HTTP - is a transpotation protocol

* SKELETON - will get method details ~~[illegible]~~ from SOAP req. it invokes client method details from actual service class & get return values & store in SOAP response ~~class~~

* WSDL generation tool is one class.
* Stub genon tool is also predefn class
* SKELETON is also a prefn class

* JAVA → Sun ~~should~~ should provi WSDL & SKELETON
↳ C# → microsoft shoul pro WSDL & SKELETON.
↳ WSDL SKELETON } API (Provided by Java) } Server

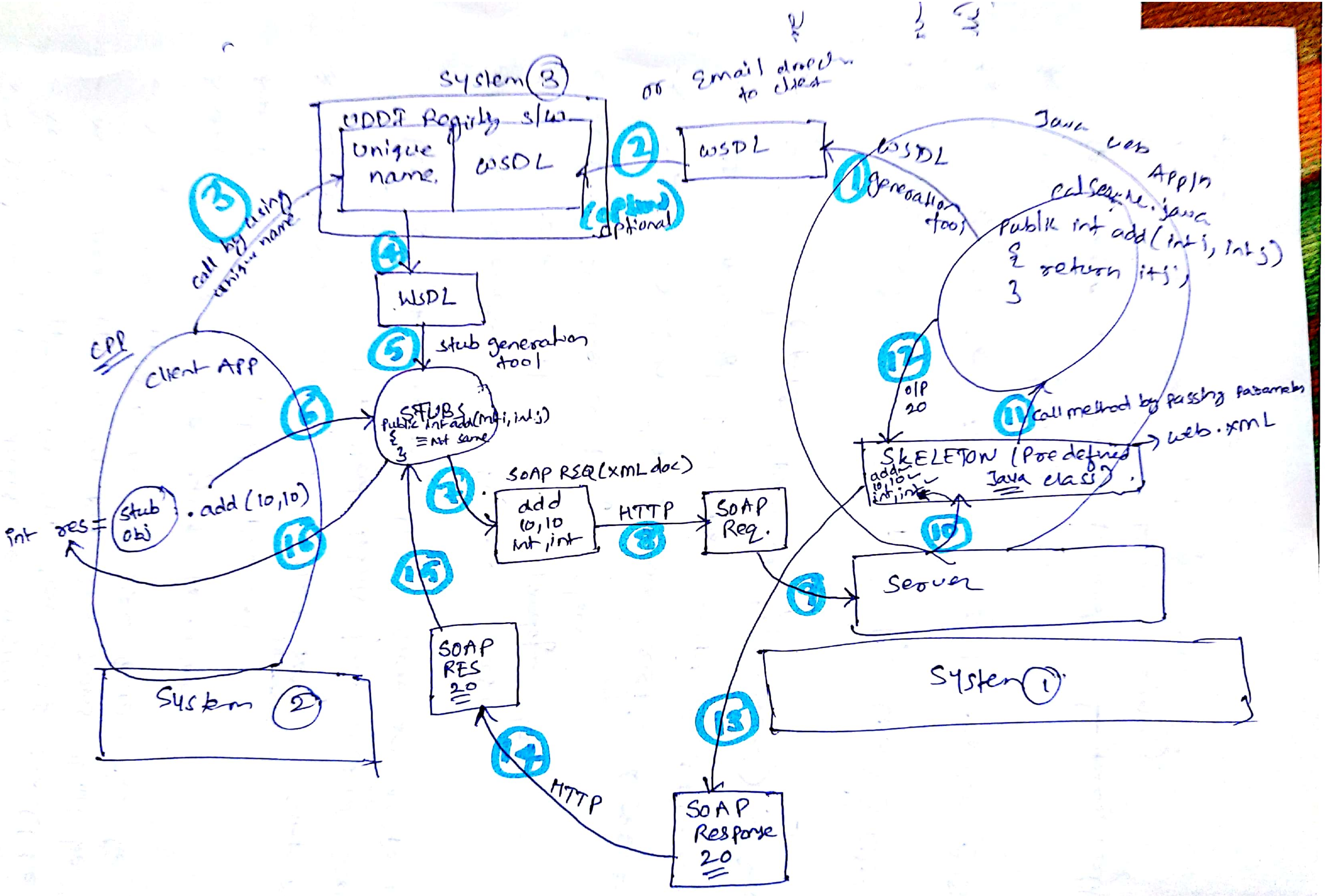
↳ STUBS → } ~~Java~~ API (Provi by Java) } client

↳ All should be in Java web service API's

* Java Provide four API's.

JAX-RPC — Remote Procedure ~~langue~~ call
JAX-M — Java API for messaging
JAX-WS — Java API for web services
JAX-RS - Remote serv

Scanned by CamScanner



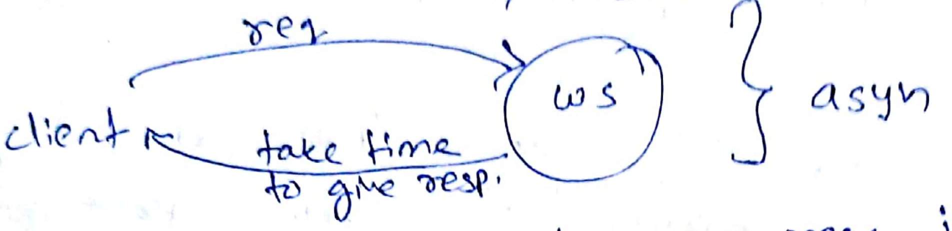
Scanned by CamScanner

* Java WebService API's

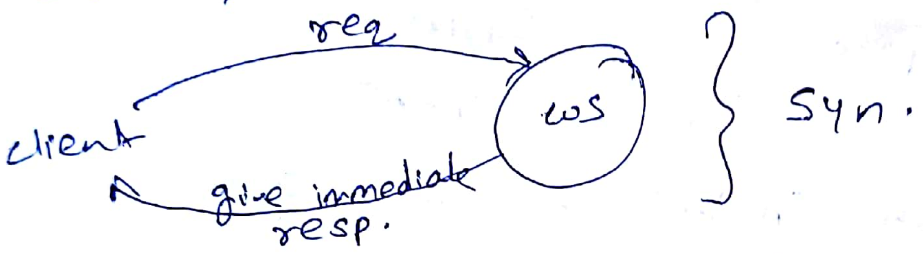
↳ Four API's → SOAP

1) JAX-RPC → Syn W.S service — Jdk1.4
2) JAX-M → Asynchronous W.S services
3) JAX-WS → Syn W.S services — Jdk1.5
4) JAX-RS → REST → Jdk1.6

} JAVA API's.



↳ Asyn :- whenever client make req. if it is asyn web services, client never wait for response.



↳ syn :- give immidi response for given req.

* Syn web services are two types

1) SOAP based WS { JAX-RPC, JAX WS }
2) RESTful WS { JAX-RS }

* JAX-RPC (Java API for XML-Remote Procedure call).

↳ It's a specfication given by sun along with Jdk1.4.

↳ using this API we can develop both SOAP based web services or SOA ~~based~~ web service clients.

* Implementation → sun Implementation (download seprately)

① → JAX-RPC-SI - SEA - API

② → AXIS 1 → Apache foundation

Scanned by CamScanner

3) Weblogic Implementation — BEA (now oracle).

4) Webspehere Impl^{n} — IBM } 'WSDL + SKELETON

5) Jboss Impl^{n} — REDHAT }

* JAX-WS API

download seprate Jar.

→ 1) JAX-WS-RI (Refrence Impl^{n})

2) Metro — Download Jar.

3) Axis 2 → Apache foundation (not support Spring)

4) Apache CXF → Apache fun^{n} (support Spring)

5) Weblogic Imp^{n} — BEA

6) Web speher — IBM

7) JBOSS — REDHAT

8) Glassfish (metro) Sun micro system

* JAX-RS (RESTful Services)

1) JERSEY Imp^{n} → Jar download

2) RESTEASY — REDHAT

3) REST LET — Jerome Loavel

4) Apache CXF — Apache foundation (support spring)

5) Apache wink — Apache fun^{n} (Not support)

* JAX-M / JMS (Java messaging services)

Scanned by CamScanner