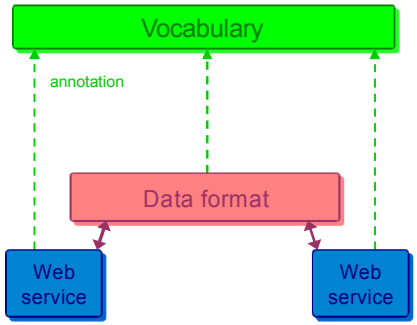
HOW TO PROVIDE INTEROPERABLE BIOINFORMATICS TOOLS

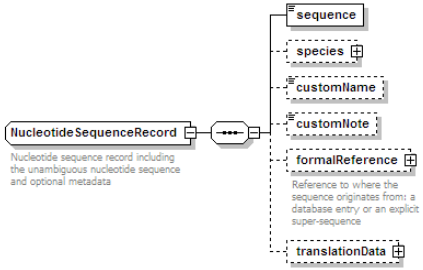
Matúš Kalaš
EMBRACE Course
CBS, Kgs. Lyngby
June 2nd, 2010



Part 1:
DESIGN-CENTRIC DEVELOPMENT ACCORDING TO EMBRACE

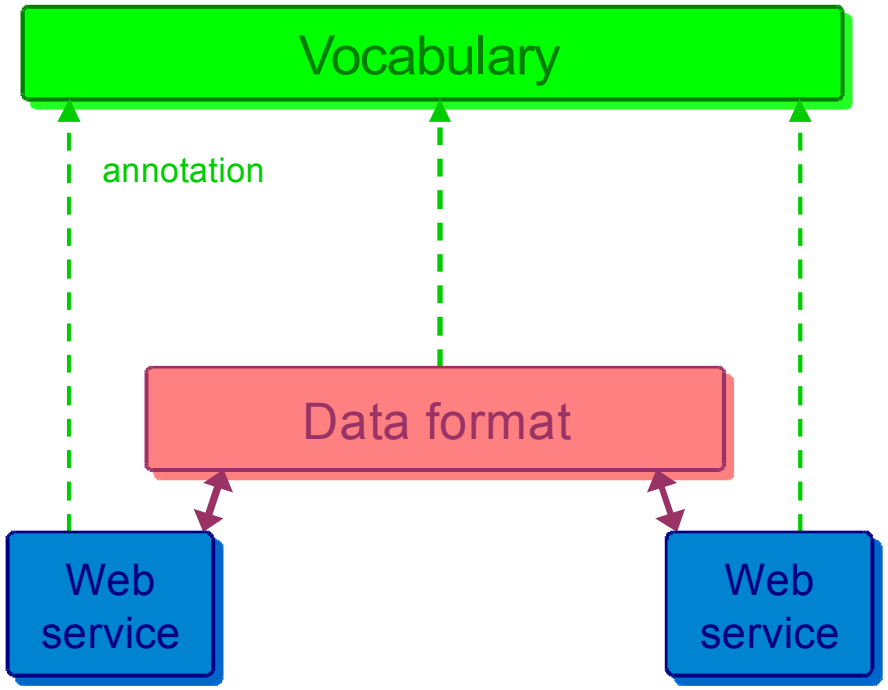
```
<xsd:import
   namespace ="http://..."
   schemaLocation ="http://..."
/>
```

Part 2:
USE OF EXTERNALLY DEFINED DATA TYPES



Part 3:
INTRODUCTION TO BioXSD

Part 1:
DESIGN-CENTRIC DEVELOPMENT ACCORDING TO EMBRACE



What is your motivation for doing Web services?

What is your motivation for doing Web services?

programmatic access
to remote
data & computing resources

What is your motivation for standardisation?

What is your motivation for standardisation?

standardisation = interoperability

enables users to discover services

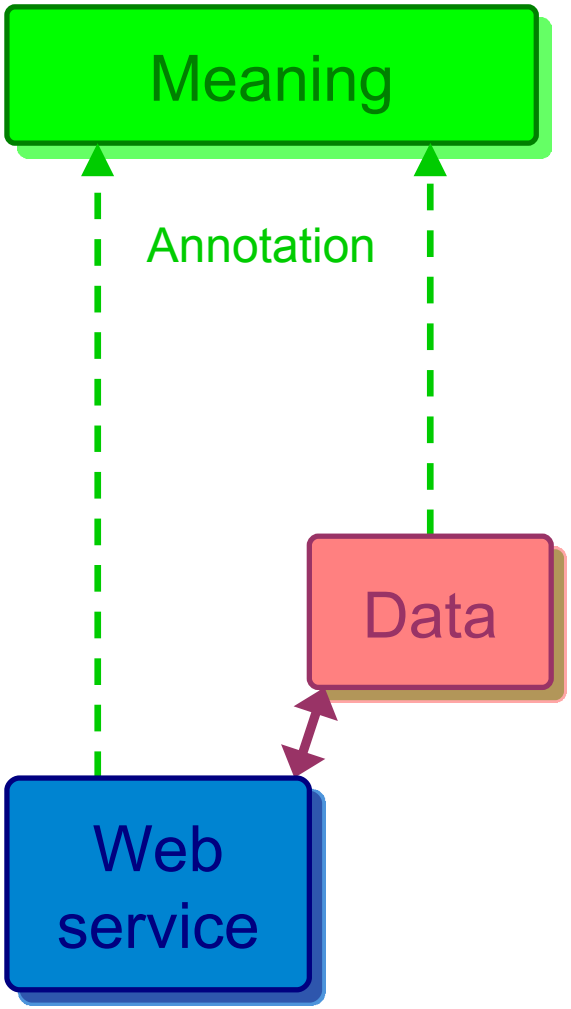
enables users to use services

discover operations & parts of data
operate on standardised input/output formats

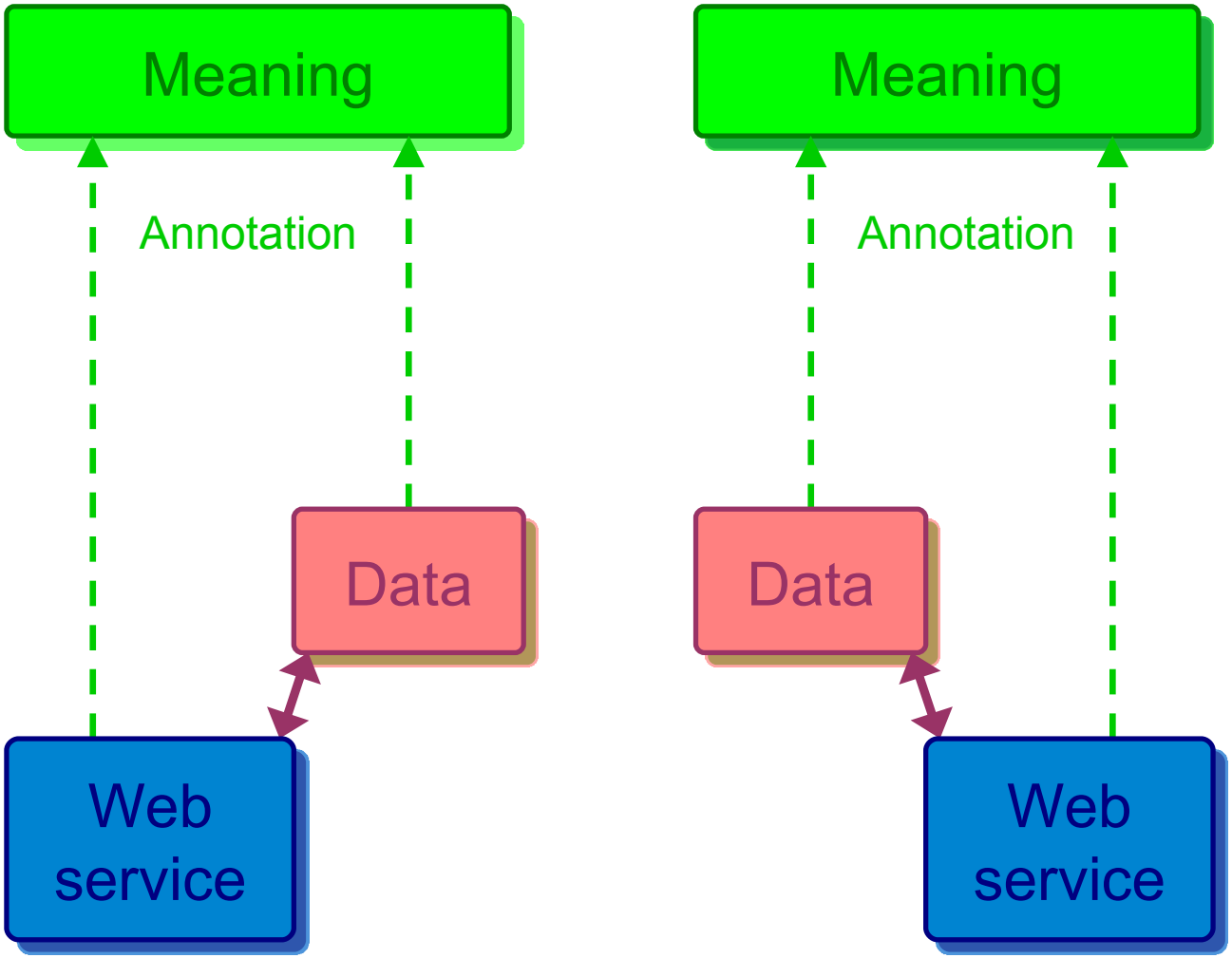
enables users to compose services

mix-and-match services smoothly into workflows/pipelines

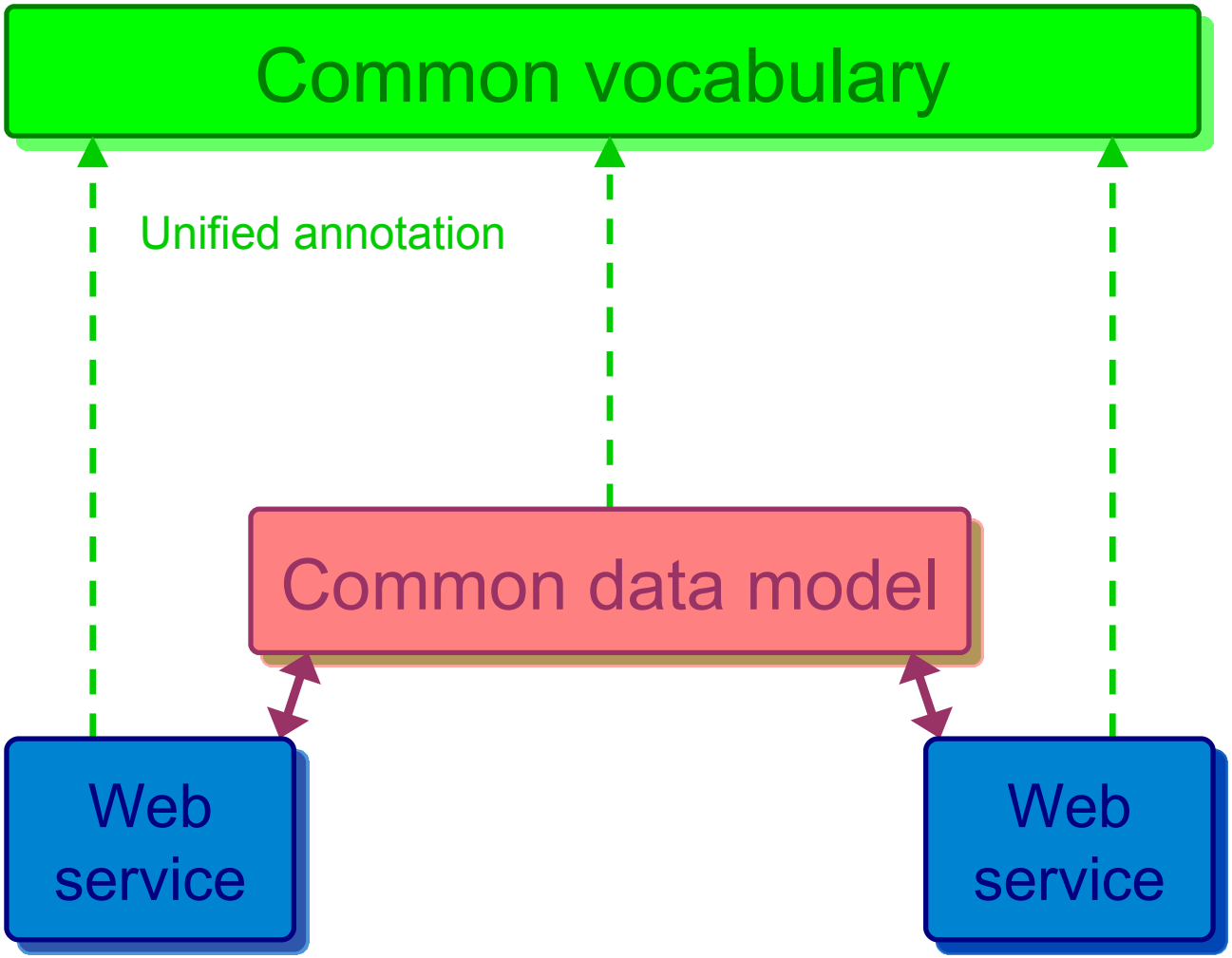
Content of tomorrow's hands-on:



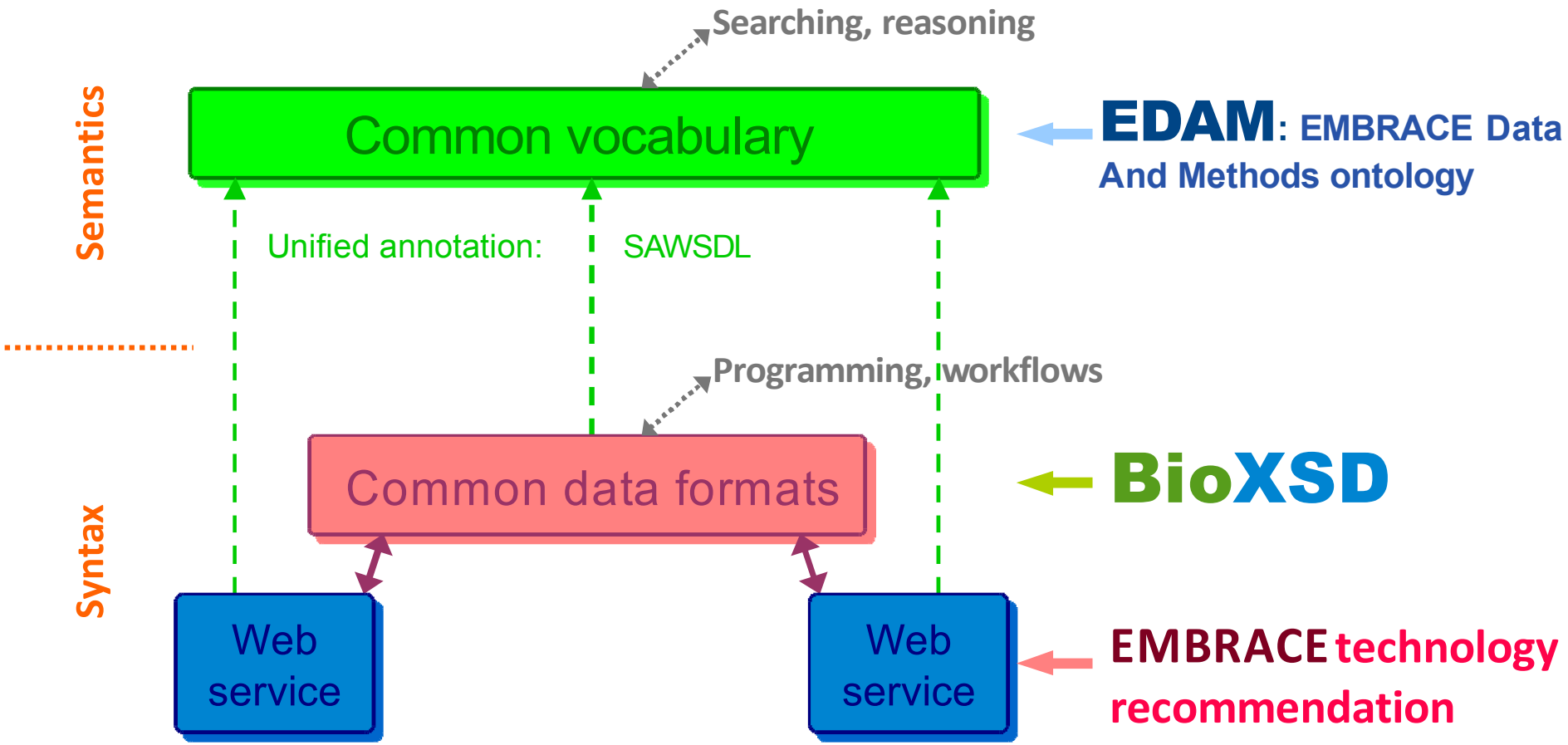
Keeping in mind the interoperability:



Keeping in mind the interoperability:



EMBRACE recommendation for providing interoperable bioinformatics tools & data sources



The recommended design-centric development:

1. **write WSDL document**
2. **use WS-I standard & doc-lit wrapped SOAP binding, design invocation pattern**
3. **services and operations:**
   - name well
   - annotate by EDAM
4. **structure the input and output data into XML**
   - write XSD complexTypes
   - name well
   - annotate by EDAM
   - use an external common data model when & where applicable (BioXSD complexTypes)
5. **restrict values of atomic pieces of data**
   - restrict XSD simpleTypes
   - use an external common data model when & where applicable (BioXSD simpleTypes)
6. **implement the Web service**
   - implement the request handler
   - connect to the tool function (via an API or calling an executable)
7. **test the Web service & its interoperability**
   - example client programs

The content of tomorrow's hands-on:

1. write WSDL document
2. use WS-I standard & doc-lit wrapped SOAP binding, design invocation pattern
3. services and operations:
   - name well
   - annotate by EDAM
4. structure the input and output data into XML
   - write XSD complexTypes
   - name well
   - annotate by EDAM
   - use an external common data model when & where applicable (BioXSD complexTypes)
5. restrict values of atomic pieces of data
   - restrict XSD simpleTypes
   - use an external common data model when & where applicable (BioXSD simpleTypes)
6. implement the Web service
   - implement the request handler
   - connect to the tool function (via an API or calling an executable)
7. test the Web service & its interoperability
   - example client programs

Part 2:
USE OF EXTERNALLY DEFINED DATA TYPES

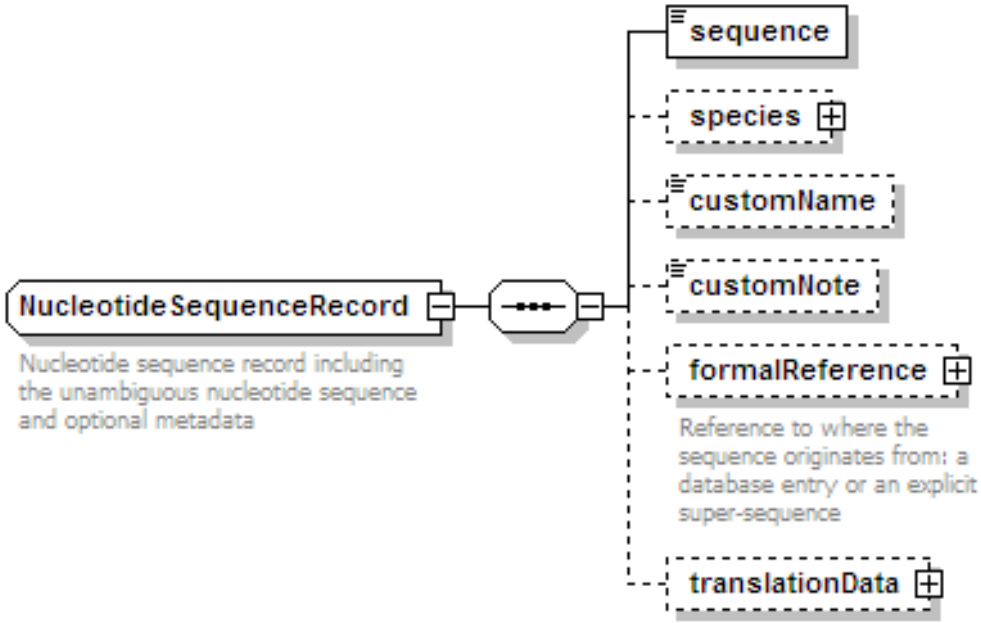
```
<xsd:import
    namespace="http://..."
    schemaLocation="http://..."
/>
```

The pattern for using external data types in WSDL:

```
<wsdl:definitions
    name="MyService"
    targetNamespace="http://www.myOrg.org/myService"
    xmlns="http://www.myOrg.org/myService"
    xmlns:exter="http://www.externalOrg.org/externalSchema"
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    .
    .
>

    <wsdl:types>
        <xs:schema targetNamespace="http://www.myOrg.org/myService">

        <xs:import
            namespace="http://www.externalOrg.org/externalSchema"
            schemaLocation="http://www.externalOrg.org/someLocation/externalSchema.xsd"
        />

        <xs:element name="runMyService">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="argument1" type="exter:ExternalType"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>

        .
        .
        .



        </xs:schema>
    </wsdl:types>
```

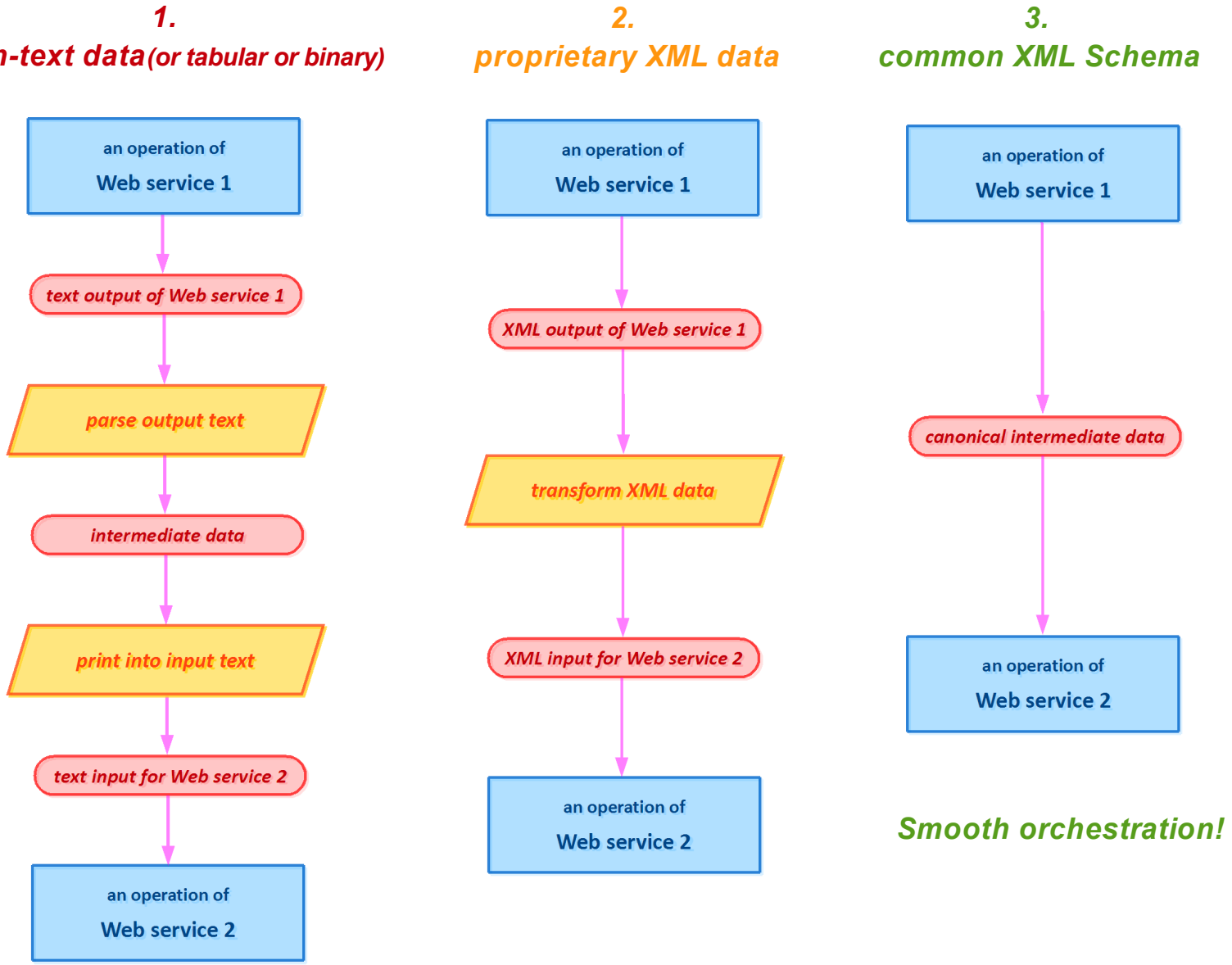
Part 3:

INTRODUCTION TO BioXSD



Motivation for users

What does BioXSD offer to service users?



Motivation for providers

What does BioXSD offer to service providers?

ready-made building blocks

Main BioXSD types:

SimpleTypes:

Accession(s)

NucleotideSequence
AminoacidSequence
GeneralNucleotideSequence
GeneralAminoacidSequence
Biosequence

helper types:
Name, FreeText
Uri
Integer(s), Decimal(s)
…a few more

Main BioXSD types:

SimpleTypes:

Accession(s)

NucleotideSequence
AminoacidSequence
GeneralNucleotideSequence
GeneralAminoacidSequence
Biosequence

helper types:
Name, FreeText
Uri
Integer(s), Decimal(s)
…a few more

ComplexTypes:

NucleotideSequenceRecord
AminoacidSequenceRecord
GeneralNucleotideSequenceRecord
GeneralAminoacidSequenceRecord
BiosequenceRecord

..SequenceAlignment

AnnotatedSequence

DatabaseReference, EntryReference
OntologyReference, OntologyTerm
Species, SequenceReference, Method

helper types:
Score, SequencePosition(s)
…some more

BioXSD.org