情報ネットワーク論1
アプリケーションプロトコル

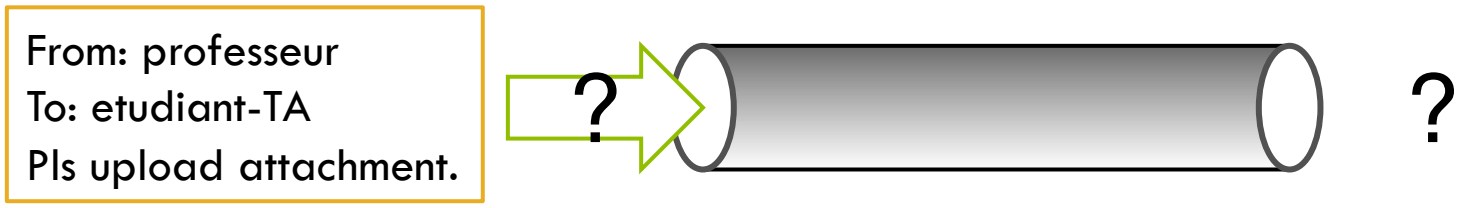
2012/5/23

門林　雄基

NAIST 奈良先端科学技術大学院大学

アプリケーションプロトコルの挑戦

- どのように情報を送信するか
  - TCPを使用することはできるか?
- どのように通信相手と相互通信を行うか？
- どのように通信相手を見つけるか?



エンコーディングのやり方

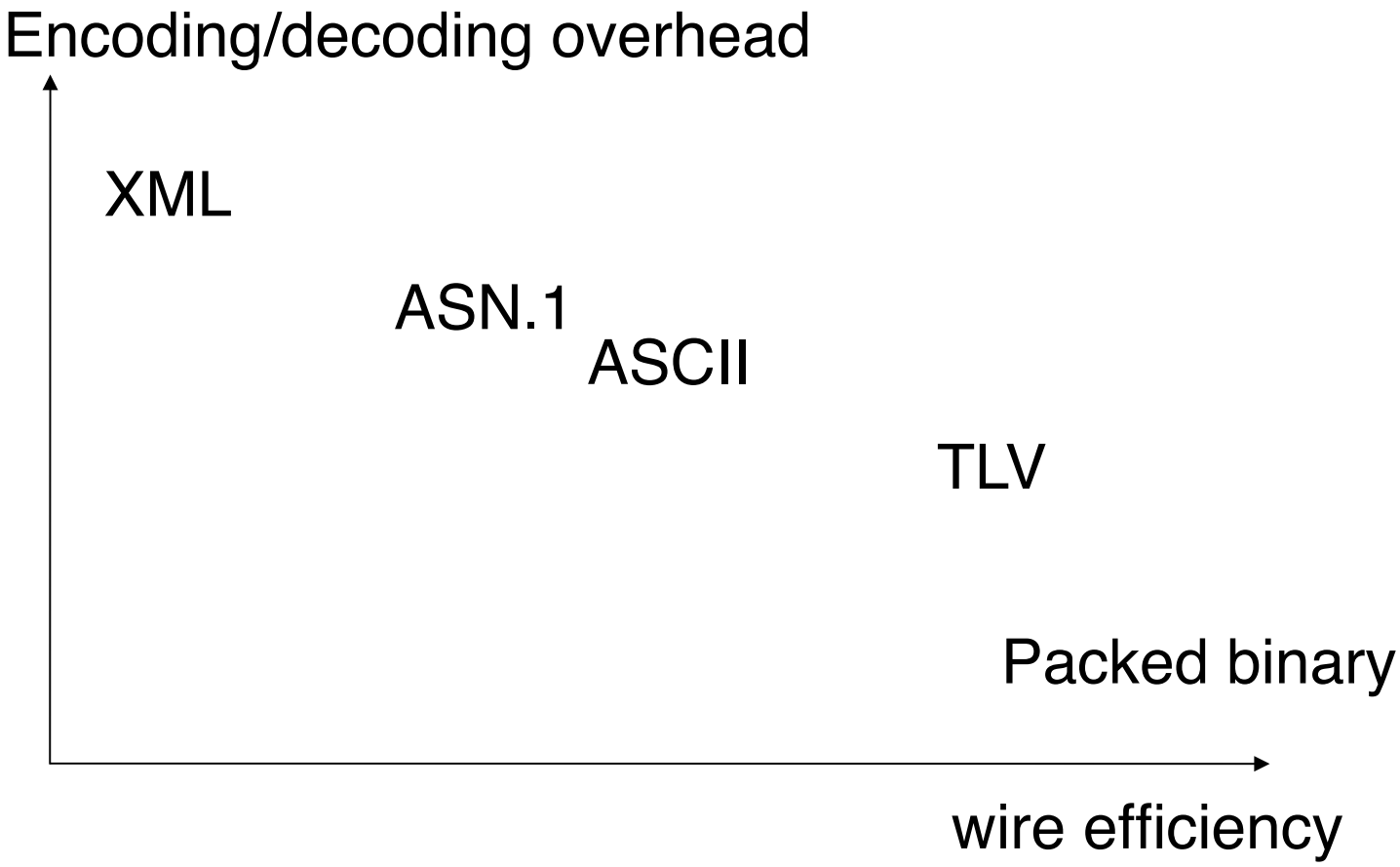
- 数字の1はさまざま方法で表現される:
  - 00000001
  - 65 (ASCII code)
  - Type=1 Length=2 Value=1

→ プレゼンテーション層

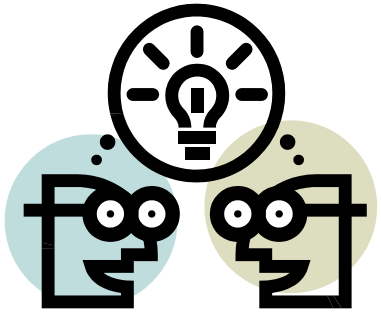
プレゼンテーション層でのエンコーディング方法の確立

- Binary
  - 00000001
- ASCII
  - 65
- TLV
  - Type=1 Length=2 Value=1
- ASN.1
  - → next lecture (network management)
- XML

データ表現に関するトレードオフ



Q&A



 12/05/23

Encoding application protocols

Design choices

- Packed binary
- ASCII
- TLV
- XML
- ASN.1

… impacting:

- Efficiency
- Extensibility
- Readability
- Ease of definition
- Ease of parsing
- Ease of mixing
- Ease of validation

Packed binary

- ビットの境界を定義
- 各ビットの意味を定義

- IP header (RFC791), TCP header (RFC793)
- アプリケーション層では同じ技術を採用することができる,
  例)センサーネットワークの技術

16bit source port			16bit destination port
32bit sequence number			
32bit acknowledgment number			
4bit hlen	reserved	flags	16bit window size
16bit TCP checksum			16bit urgent pointer

20 octets

ASCII

- ASCIIコードのバイトストリームをプロトコルとして考える
  - ポイント: 人間が読めるようにする
- インターネットでのASCIIプロトコル
  - 機械が解析可能なコードと人間が可読可能な文字列の混合
- FTP (RFC959), SMTP (RFC2821) etc.
  - セッション開始につづく3桁の数字のステータスコード
  - 引数につづく4文字コマンド

FTP: File Transfer Protocol

01.204.01754: 220 ftp.isi.edu NcFTPd Server (free educational license) ready.
76.020.00021: USER anonymous
01.204.01754: 331 Guest login ok, send your complete e-mail address as password.
76.020.00021: PASS -wget@
01.204.01754: 230 Logged in anonymously.

76.020.00021: SYST
01.204.01754: 215 UNIX Type: L8
76.020.00021: PWD
01.204.01754: 257 "/" is cwd.
76.020.00021: TYPE I
01.204.01754: 200 Type okay.
76.020.00021: CWD /in-notes
01.204.01754: 250 "/in-notes" is new cwd.

SMTP: Simple Mail Transfer Protocol

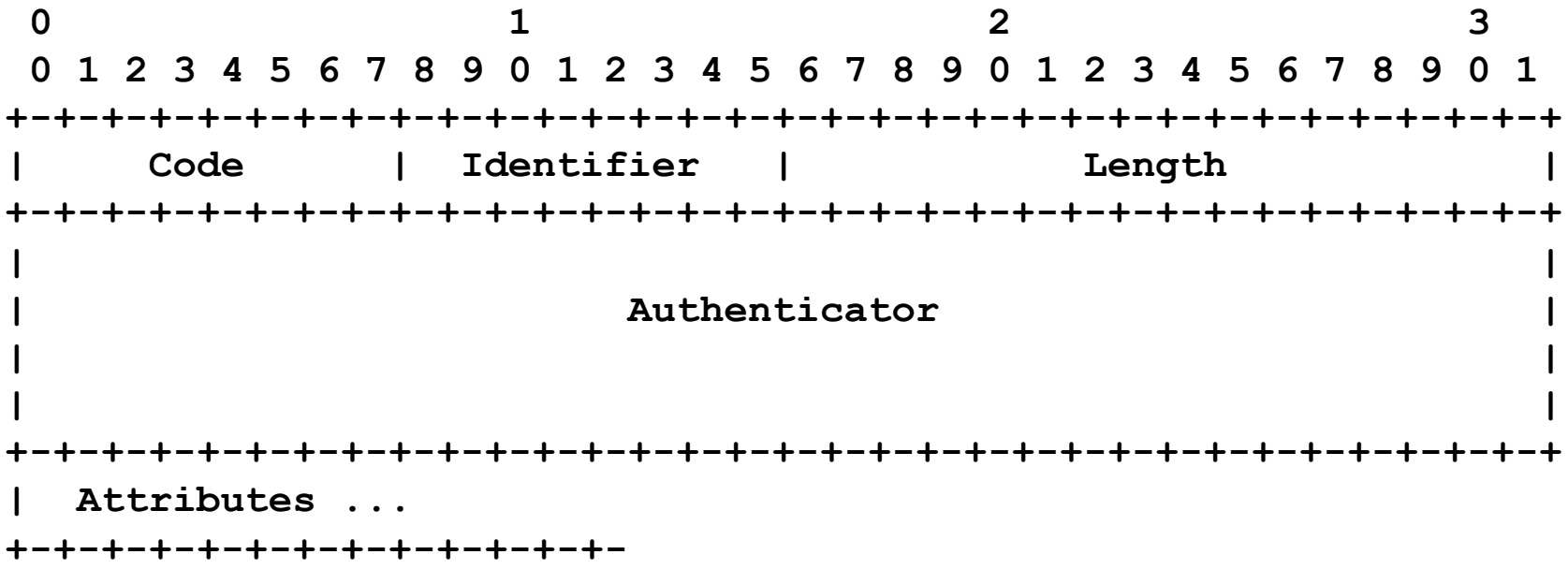

```
1758: 220 ns.iij-mc.com ESMTP Sendmail 8.9.3p2+3.1W/3.7W/ns; Fri, 30 May 2003 00:15:43 +09
0025: EHLO mf.aist-nara.ac.jp
1758: 250 ns.iij-mc.com Hello 168.pool3.ftthtokyo.att.ne.jp [165.76.67.168], pleased to meet you
0025: MAIL FROM:<youki@is.aist-nara.ac.jp> SIZE=1524
1758: 250 <youki@is.aist-nara.ac.jp>... Sender ok
0025: RCPT TO:<bunji@iij-mc.com>
1758: 250 <bunji@iij-mc.com>... Recipient ok
0025: DATA
1758: 354 Enter mail, end with "." on a line by itself
0025: Received: from mf.aist-nara.ac.jp (localhost [127.0.0.1])

3 +0900 (JST)
```


TLV

- タイプ，長さ，数値
  - 利点:
    コンパクトで，伸長性がある
  - 欠点:
    すべてのタイプはプロトコルによって定義されるべきである
- RADIUS (RFC2138), OSPF (RFC2328) etc.
  - TLVのビット表現

RADIUS Packet Format

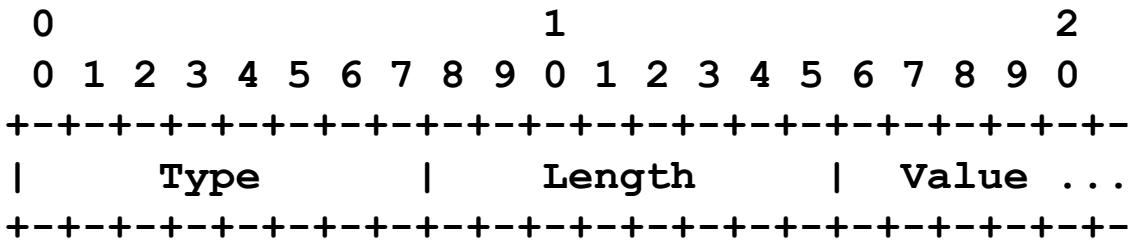
- (RFC2138) 3. Packet Format



Code:

1	Access-Request
2	Access-Accept
3	Access-Reject

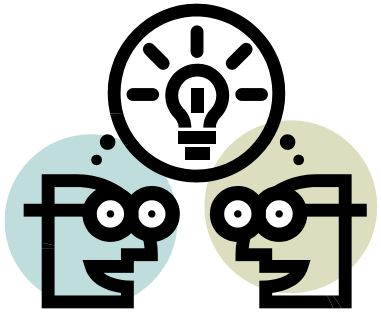
RADIUS Attribute TLV



Type

1	User-Name
2	User-Password
3	CHAP-Password
4	NAS-IP-Address

Q&A

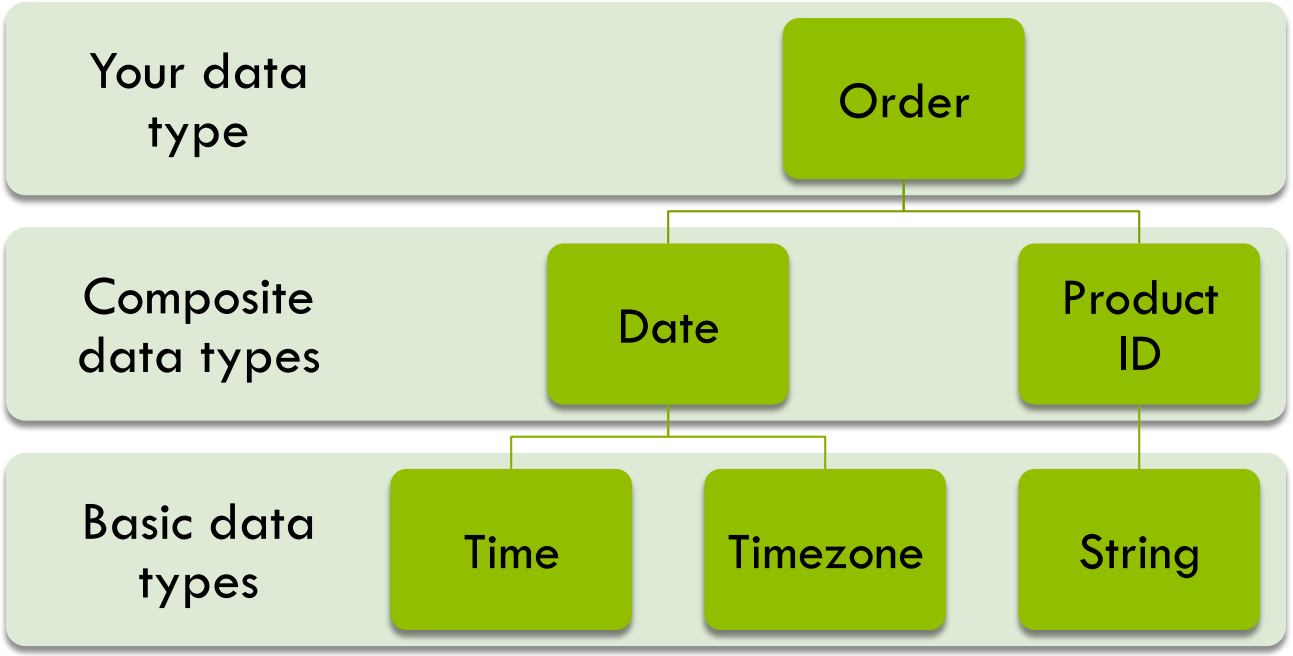


 12/05/23

XML: Extensible data types

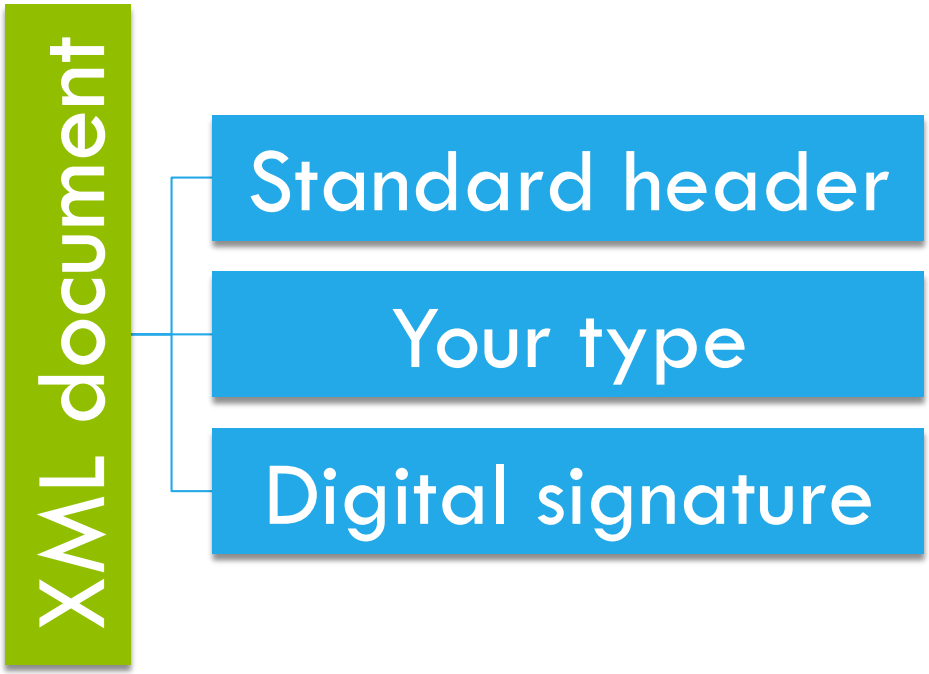
- データタイプ(静的/動的な型付け)
- 名前空間の定義
- それ自身で記述されるデータ
- 潤沢なツールとライブラリ

XML: Reuse of data types



- 自分で型を定義するために”Date”タイプを自分で再定義する必要はない

XML: Modular data types



- すでに定義されている文章の構造や電子署名などを再利用出来る

XML message format

- <?xml version = "1.0"?>
- <tag attribute="value">
  <another-tag another-attribute="value" />
- </tag attribute="value">

XML名前空間を利用した名前空間の利用


```
<?xml version='1.0' encoding='UTF-8'?>
<soap:Envelope xmlns:soap='http://schemas.xmlsoap.org/soap/envelope/'
xmlns:xsi='http://www.w3.org/1999/XMLSchema-instance'
xmlns:xsd='http://www.w3.org/1999/XMLSchema' xmlns:soapenc='http://
schemas.xmlsoap.org/soap/encoding/' soap:encodingStyle='http://
schemas.xmlsoap.org/soap/encoding/'>
 <soap:Body>
  <n:getQuoteResponse xmlns:n='urn:xmethods-delayed-quotes'>
  <Result xsi:type='xsd:float'>7.92</Result>
  </n:getQuoteResponse>
 </soap:Body>
</soap:Envelope>
```


XMLでのデータタイプの記述: XML Schema


```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:xsi="http://www.w3.org/1999/
XMLSchema-instance" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/
soap/encoding/" xmlns:xsd="http://www.w3.org/1999/XMLSchema“
SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
 <SOAP-ENV:Body>
  <namesp1:getQuote xmlns:namesp1="urn:xmethods-delayed-quotes">
  <symbol xsi:type="xsd:string">AKAM</symbol>
  </namesp1:getQuote>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```


XML: 他の機能

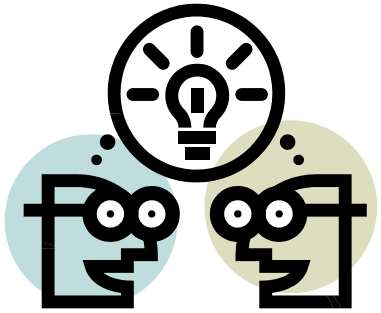
- 自己記述性: XML Schema
- 翻訳ツール: XPath, XSLT, etc.
- 潤沢なAPI: DOM, SAX, etc.

- メッセージの完全性と機密性:
  - XML Digital Signature
  - XML Encryption

プレゼンテーション層のまとめ: 鳥瞰図

Approach	Fixed	Extensible	
	Statically typed		Dynamically typed
binary	Packed binary	TLV	ASN.1
string		ASCII	XML

Q&A



 12/05/23

どのように通信相手との相互通信を行うか?

- 誰がイニシエータになるのか?
- 誰が応答者になるのか?

- どのような種類の情報が送信されるのか?
- 任意の状態遷移?
- 本質的には, プロトコルFSM(有限オートマトン) デザインを行う
- プロトコルFSMはデザイン時に気を付けないと, デッドロックを起こしてしまう

プロトコル属性の保証

- 生存性
  - なにかいい事が最終的に発生する
  - e.g., データ転送が完了する
- 安全性
  - なにか悪いことが起きる
  - e.g., 欠乏
- → モデル検査

モデル検査を用いたプロトコル属性の保証


```
chan q[2] = [MaxSeq] of { byte, byte };   /* message passing channel */

active [2] proctype p5()     /* starts two copies of proctype p5 */
{ byte NextFrame, AckExp, FrameExp, r, s, nbuf, i;
  chan in, out;
  in = q[_pid];
  out = q[1-_pid];
  xr in; xs out;          /* partial order reduction claims */

  do
  :: nbuf < MaxSeq ->    /* outgoing messages */
       nbuf++;
       out!NextFrame,    (FrameExp + MaxSeq) percent (MaxSeq + 1);
       inc(NextFrame)

  :: q[_pid]?r,s ->        /* incoming messages */
       if
       :: r == FrameExp ->
            printf("MSC: accept percentd\n," r);
            inc(FrameExp)
       :: else /* ignore message */
       fi;
       do
       :: ((AckExp <= s) && (s < NextFrame))
       || ((AckExp <= s) && (NextFrame < AckExp))
       || ((s < NextFrame) && (NextFrame < AckExp)) ->
             nbuf--;
             inc(AckExp)
       :: else -> break
       od
```

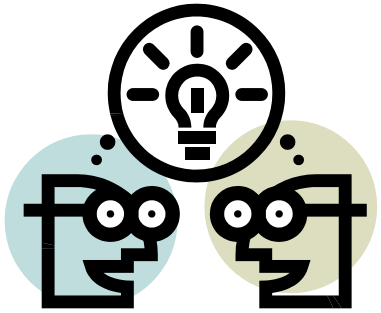
Appendix C: Verification model of a sliding window protocol

For further study:

- **LTL**
- **Model checking**

Source: The model checker SPIN, IEEE Trans. on Software Engineering, May 1997

Q&A



 12/05/23

どのように通信相手を発見するか?

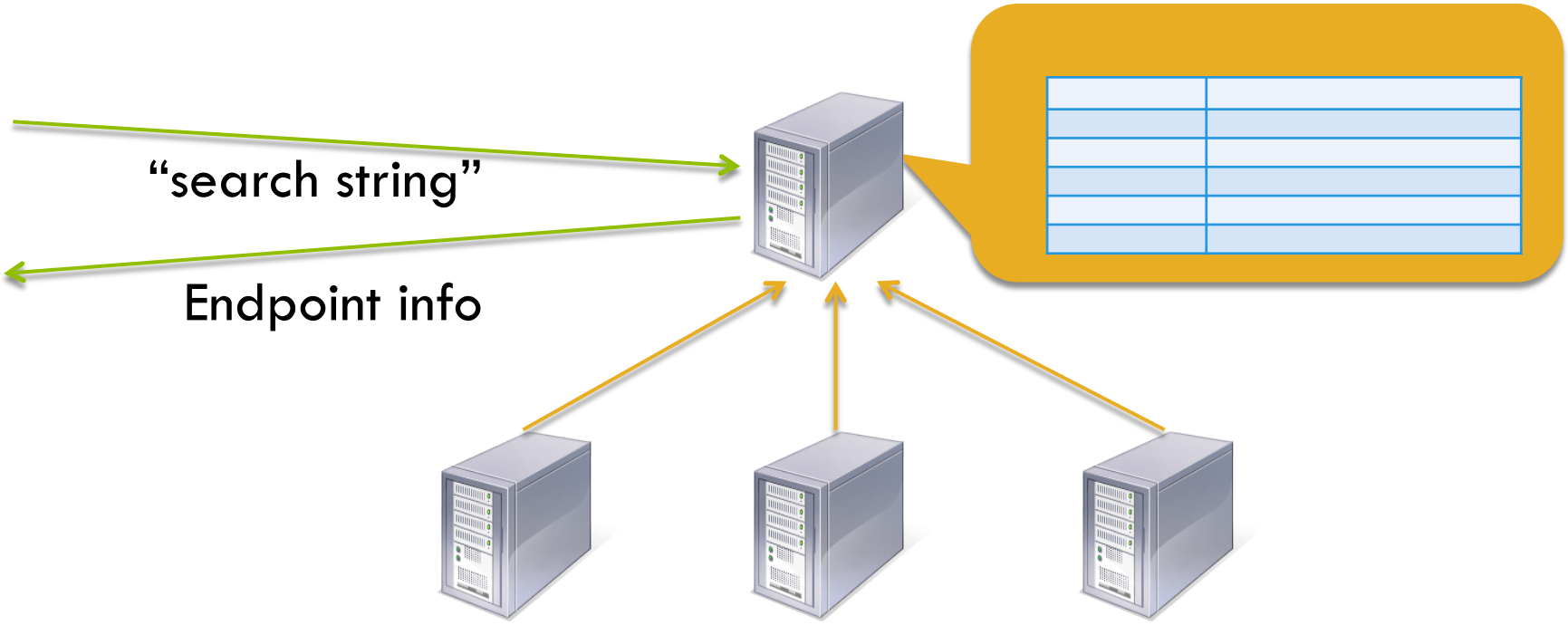
- 1.相手を予め知っている
- 2. 中央権威サーバが相手を知っている
- 3. 相手は情報を処理している人でなければならない.

エンドポイントについて事前知識

- e.g., by Uniform Resource Locator (RFC1738)
  - アプリケーションプロトコルの型
  - TCPポート番号
  - Host name
  - etc.

```
      http://example.com:8080/over/there?name=ferret#nose
       \_/   \______________/\_________/ \_________/ \__/
        |           |            |            |        |
     scheme     authority       path        query   fragment
```


エンドポイント発見のための中央集権システム



- 多くのアルゴリズムは1000を超えるサーバで規模拡張可能
- 今後の課題: NSDI, SIGCOMM papers

情報の種類からエンドポイントを発見する

- リソースの関連からサービスの発見

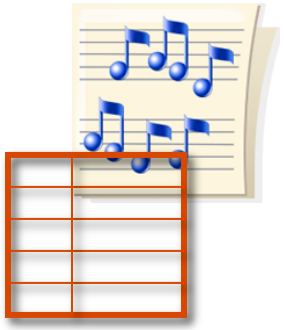
Resource identifier

↓

Resolver

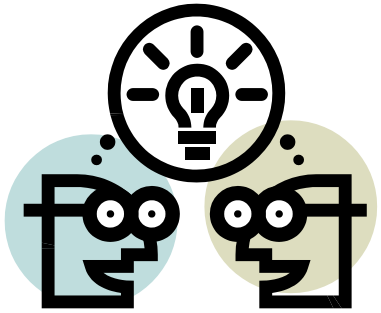
↓

Service endpoint identifier
Capabilities



?

Q&A



 12/05/23

まとめ: アプリケーションプロトコル

- どのように情報を送信するか?
  - プレゼンテーション層に依存する
- どのように通信相手と相互通信するか?
  - 自分のプロトコルを動かすことができるが，注意が必要
- どのように相手を発見するか?
  - 3つの表現パターンがある

