ビジネスインフォマティクスの概観と体系
(筑波大学 EDP-entry「インターネットのビジネスインパクト」＃4)

久野　靖*

2005.8.19

1　今回のストーリー

□ ビジネス・インフォマティクスとは?　なぜ必要か?

□ ビジネス・インフォマティクスの課題例

□ ビジネス・インフォマティクスの体系・全体像

□ 代表的なドメインのクローズアップ

- → ネットワーク、WWW

2　ビジネス・インフォマティクス (BI) とその必要性

□ 従来の情報系の学問体系 --- JABEE の認定プログラム (情報系) を見ると:

- コンピュータ科学 (CS) --- プログラミング、言語、OS
- コンピュータ工学 (CE) --- ハードウェア、システム設計
- ソフトウェア工学 (SE) --- システム分析、ソフト開発
- 情報システム (IS) --- 情報システム設計、開発、運用

□ その道の (深い) 専門家→他の分野の人がその素養を獲得するのは困難

□ 一方、今日ではビジネスに情報システムは不可欠

- 20 世紀:　情報システムはビジネスを効率化/サポート
- 21 世紀:　情報システムなしには存在し得ないビジネスが主流に

□ →「情報のことは専門家が」では済まない世の中に

- ではどうするか?

□ 解 1:　情報の専門家が起業し経営者に→確かに存在

- それが唯一解では困る。ビジネス専門家も必要なはず

□ 解 2:　ビジネスパーソンであっても情報の基本原理とメカニズムを熟知→ビジネスインフォマティクスの目標

- そんなことが可能か?　→だからこそチャレンジ!

□ 情報の専門家→今日では専門の範囲が極めて狭い

- 情報技術の範囲は極めて広くかつ急速に変化
- 専門技術者、研究者なら専門だけでよいが…
- ビジネスに情報技術を駆使する場合→自分の専門範囲だけでは済まない。他分野のことを「知らない」まま進むことは危険

□ 必要な全範囲をカバーする理解→ビジネスインフォマティクスの目標

3　ビジネス・インフォマティクス (BI) の定義

□ 情報の非専門家や特定範囲の専門家が、

□ 必要に応じ情報各分野の専門家と協働する上で必要な、

□ 原理理解と思考体系獲得を可能とするための、

□ 知識/技術体系化手法の総体

□ →ビジネス・インフォマティクス (BI)

4　ビジネス・インフォマティクスの課題例

□ 次のような課題を考えてみる

*筑波大学ビジネス科学研究科

- 地球の裏側 (ブラジル) に支社がありそこの在庫管理を当地のサーバでやっている
- システム管理、メンテナンス、トラブル対応等が日本と全く別なので問題が多い
- そこで、本社 (日本) のサーバで支社の在庫管理もするのはどうか?

□ Q:これだけ聞いて「アリ」「ナシ」どっち?

□ Q:このプランの可否を判断するのには何が分かっている必要?

- ネットワークの伝達時間は実用範囲内なの?
- ソフトウェア開発の手間/コストは?
- システムの可用性や信頼性は?
- セキュリティは?

□ それぞれについて根拠を持った判断ができるかどうか?

□ これらのことを「専門家に聞いて受け売りする」のではダメ

- 自分で考え納得した上でなければ責任も取れない
- 考えた上で専門家と議論して詰めるのなら正しい
- アイデアは非専門家の方が出せる (ただし feasible でないと…)

□ これらを自力で考える上で必要な土台の構築＝ BI の目標

5 ビジネス・インフォマティクスの体系

□ 従来の「○○が分かる」的解説本の駄目なところ…

- 個別のドメインの話だけしても「例示」「現象」にとどまる←本一冊で素人に説明するとしたらそれしかないから
- そのようなものを多数取り入れても「薄皮饅頭」

□ BI の主張: 最初は敷居が高くても土台から積み上げる方が早道

□ [1] コンピュータの原理→プログラミング言語→ OS/プログラムの実行

- 「プログラムを動かす体験」は必須
- 「プログラムをゼロから書く」訓練は時間的に難しい
- サンプルを持って来て/動かし/直して観察 (理科の実験モデル)

□ [2] データ記憶→ファイルシステム→データベース

- 裸の記憶装置→ OS による機能付加→ファイル
- データの共有/定型化→データベース/DBMS
- DB の用途→ Web サービス、データウェアハウス

□ [3] コンピュータとの対話→ユーザインタフェース→ CUI/GUI/Web

- 「対話」が持つ意味→人間の能力を引き出す
- 人間側の持つ特製、限界→認知的側面
- コンピュータ側の選択肢→多くの可能性 (普通の GUI 以外)

□ [4] ネットワークの原理→ネットサービス→ Web/Web アプリ

- パケット→経路制御/伝送制御→自在な通信媒体
- インフラ (インターネット) →各種サービス
- 代表例: Web →基本アイデア→多様な活用

□ [5] 各種メディア→ドキュメント/マルチメディア/動的

□ [6] AI 技術→学習、推論、知識表現

□ [7] データ→データマイニング、自然言語

□ [8] 情報社会→リテラシ、セキュリティ、ネット社会

□ [9] 情報システムの分析/設計/開発→ソフト工学/ビジネスプロセス

□ [X] システムズマネジメントアプローチ→研究型/他プログラムと連携

□ BI では「どのように」各ドメイン内を体系化するか?
→ 下記の問い

- (「何がどのようになっているのか?」) ←普通の知識
- 「何がもっとも本質的か?」
- 「原理的に何が可能で何が不可能か?」
- 上記 3 点に:「それはなぜか?」←もっとも重要

□ すべての「なぜ」には技術的/経済的/社会的理由

- それなしに「こうなのだ」と断言しても単なる演説

□ すべての「なぜ」に具体的に答えられるようになるための体系=ストーリを提供→ BI が提供する体系化

6 クローズアップ: ネットワークの原理と応用

□ ネットワークのドメインから具体例を抜粋

- 全体テーマ「インターネットのビジネスインパクト」ですから…
- ただしわずか数十分で抜粋なので全ての「なぜ」には対応しない
- BI プログラムの雰囲気を味わって頂ければ幸いです

□ 進め方:　各自 quiz を記入→続いてレクチャー

6.1　ネットワークとその目的

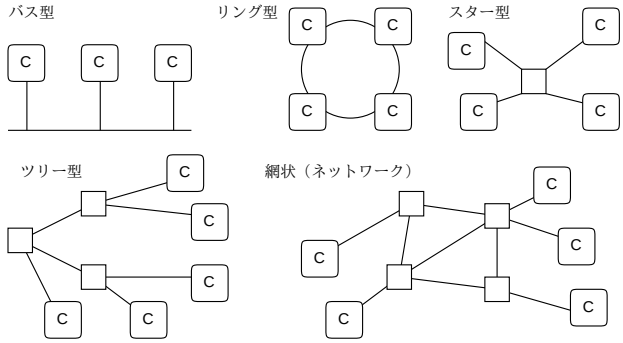
□ 計算機ネットワークを構成する目的として何がある?

- 資源の共有 --- 「ファイルサーバ」「計算サーバ」
- 信頼性 --- 1 台止まっても残りで処理
- 経済性 --- 巨大なマシンより量産品は割安
- 段階的成長 --- マシンを増やす→能力増強
- 通信媒体 --- 「距離をなくす」新たな応用

6.2　ネットワークとトポロジ

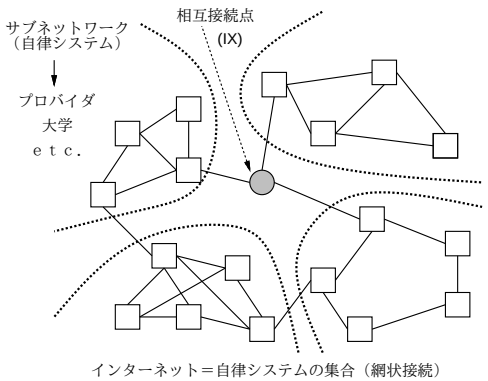
□ Q:インターネットの「つながり方」はどんな感じ?

□ 典型的なネットワークトポロジ:



□ インターネットの場合→階層構造

- LAN/各家庭/企業内/学内→プロバイダ/プロジェクト→全体



□ 「なぜ」こうなのか?

- 「誰かが計画的に」作り上げたものではない→網状
- 一元管理でない→自律システムの集合体
- 各サブネットが「善意で」相互に配送

□ 巨大な「草の根ネットワーク」がインターネットの本質

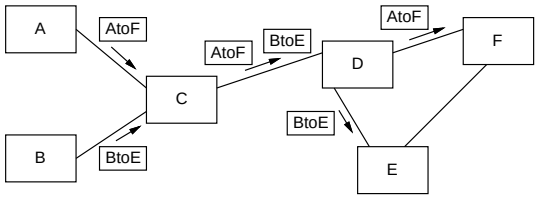
6.3　回線交換とパケット交換

□ Q:インターネット上の通信は「小包」「電話」どちらに近いか?　なぜそう思うか?

- 回線交換 (電話):　最初に伝送経路を全部確保する (ダイヤリング) →使用中はずっと確保→切断時に開放
- パケット交換 (インターネット):　データを決まったサイズ (以下) に切り分けて随時投入

□ なぜパケット交換が使われるか?

- 伝送量の増減がある→確保して使わない→無駄
- 各中継点での「確保」「開放」は大変な手間
- 「確保」されっぱなしは困る
- 中継点はできる限りシンプルに→廉価・高速



6.4　伝送制御/エラー制御

□ Q:インターネット機器のパケット損失率はどれくらい?

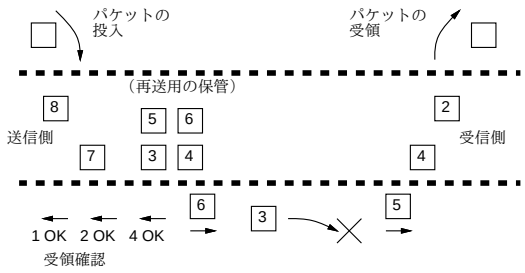
□ Q:ネット通信のエラー率は途中で通過する経路のエラー率の最悪のものによって制約される (それより良くはならない):ホント?ウソ?

□ デモ:簡単なプログラムで裸のパケットを隣のマシンに送る

- 少量のデータは当然問題なく届く
- 多くなってくると→ボロボロに落ちる

□ 品質の悪い通信路上でエラーのない伝送をするには?

- パケットに番号をつけ並べかえ
- 確認パケットが来ないと再送



□ 通信路の両端 (=コンピュータ、能力は十分ある) でサービスを提供

□ 経路上の中継点では安く早くできる範囲で単にパケットを送る

□ →インフラの変更 (=大変) なしに新しいことが始められる

6.5 経路制御と IP

□ Q:インターネット上の通信の宛先はどういう形で指定する?

□ 通信の宛先→「アドレス」

- インターネットでは「IP アドレス」(32 ビットの数値)。「192.50.17.2」のような記法で書き表す。
- 「www.google.com」みたいなのは? 「ドメインアドレス」であり、それを IP アドレスに変換して利用している。
- 理由: 人間は長い (読める) 名前がよい。中継点での処理には簡潔で効率よく扱える形式がよい。

□ Q:会社からここへ来る経路はどうやって決めたか? それと「ネットワークパケットさん」の行き先の決め方はどう違うと思う?

□ パケットの経路を決める処理→「経路制御」

- パケットは「どういう経路を通ろう」と予め計画できない
- 経路が長く、しかも頻繁に変わる可能性がある
- すべての交差点 (中継点) で宛先 IP アドレスを提示する t「あなたはこっち」と指示される (この情報もネット経由で絶えず配布更新)

□ 以上をまとめると…

- 世界中のインターネットに (直接) つながったマシンには固有の IP アドレスが割り当てられていて、
- そのアドレスを指定さえすれば途中経路がうまく選ばれ相互に行き来可能

□ 相互運用性を持つ巨大な通信基盤→インターネットの本質

- アドレス/パケットの形式→ IP(Internet Protocol) として規定。

□ Q:ブラジルまでのパケットの到達時間 (片道) はどれ位と予想?

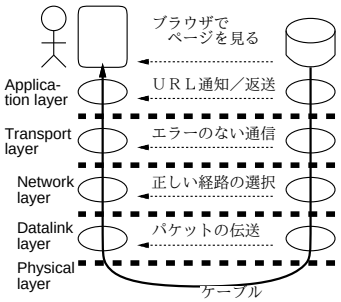
□ Q:ブラジルまでのパケットの途中中継数はいくつ位と予想?

- 実際にやってみれば分かる。
- (1) 確実にブラジルにある (ネット接続が良さそうな) サイトを探す
- (2) 「ping 宛先」→パケット往復時間が分かる (禁止してなければ)
- (3) 「tracert/traceroute 宛先」→中継点が分かる (〃)

6.6 プロトコルとプロトコル階層

□ ネットワークは多数の機能の集成→階層構造

- 下位の層の機能を利用して上位の層を実現
- 一番上には各種のサービスが来る



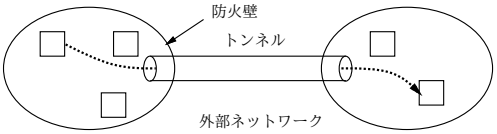
□ 各層ごとに「やりとりの約束ごと (=プロトコル) がある →全体として「インターネットプロトコル群 (TCP/IP)」

- 伝達層以下→伝送媒体毎に多数ある (パケットが通れば何でもよい)
- ネットワーク層→ IP(Internet Protocol)
- トランスポート層→ TCP(エラー制御) と UDP(裸パケット)
- その上→サービスごとにさまざま

□ Q:「ネットワークの伝送媒体としてネットワークを使う」というのは意味がない? ある? あるとすればどういう場合?

□ 解答:大あり。一般に「トンネル」と呼ばれる

- 違うプロトコルのネットワーク内を通過したい場合
- 本来は通れないところを通過したい場合←セキュリティの問題が生じることも
- 暗号化トンネル→安全に隔離された「島」どうしを結ぶ



□ その他の補足

- IP アドレスはホスト (マシン) まで識別
- サービスの識別は 1〜65535 の番号 (ポート番号) で→代表的なサービスは決まった番号を持っている
- ドメインアドレスから IP アドレスへの変換→DNS(Domain Name System) というサービスとして提供されている
- これらの番号/名前の割り当ては ICANN と呼ばれる非営利団体が管理

6.7 ネットワークサービス

□ Q:「ネットワークサービス」で思いつくものを 5 つ挙げてみて。

- 遠隔ログイン…インターネットで最初に使われたサービス
- 電子メール…同じくらい古くからあり今日もポピュラー
- ファイル転送、ネットニュース、IRC、…WWW が増えて下火に
- ファイル共有…比較的新しい
- WWW…圧倒的に今日の主流 (その上に多様な「サービス」が存在)

□ Q:ネットワークサービスを動かすマシンに必要な条件は?

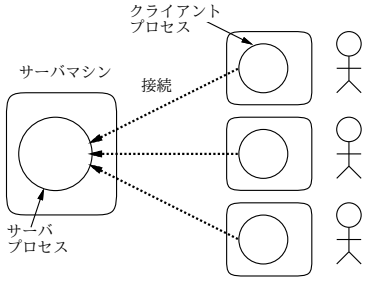
- ヒント:プログラムどうしが通信できるためには両方が同時に動く必要

□ 解答:サービスを提供する側のプログラムを「常時」動かしておける

- + 固定した IP アドレスまたはドメイン名を持ち、外部からアクセス可能

□ 「クライアントサーバ方式」

- サービスを提供する側=サーバ
- サービスを受ける側=クライアント (各ユーザが利用するマシン)



□ 以下では「WWW」を例として取り上げ検討

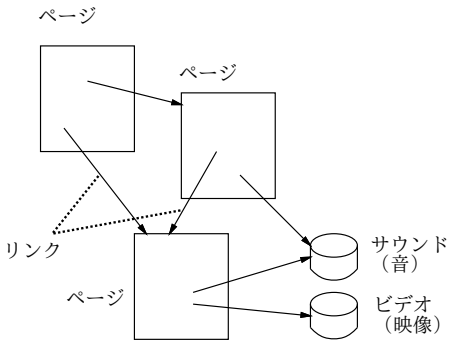
- そのサービスはどのような仕組みで作られているか?
- そのサービスでは「どんなことが」できるか?
- そのサービスを「インフラとして」何ができるか?

6.8 World Wide Web

□ Q:WWW を知らない人に「WWW とは何か」をどう説明する?

□ WWW とは:インターネット全体に分散した「ハイパーテキスト」

- ハイパーテキスト:テキスト (文書) の中にリンクが埋め込まれていて、これをたどることでさまざまな順序で読めるもの
- + 画像、音声、動画などマルチメディアに対応



□ Q:以前からインターネットはあったが、WWW の何が革新的だった?

- 必要な操作が「クリックするだけ」
- テキストに画像が混ざった情報表示
- プラットフォーム (OS) に依存しない
- 世界中のあらゆる場所の情報が指せる、無尽蔵の情報量、情報を作り出しているところから直接取れる

□ Q:WWW を作った人 (Tim Berners-Lee) の「最大の発明」は何?

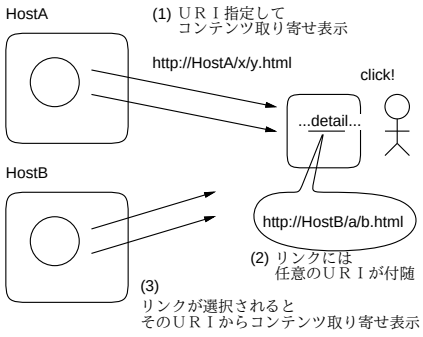
□ 解答:URI(URL)。これがあるから世界中の任意の情報が指せる

- 「http:」「mailto:」等、先頭部分の区別により、あらゆるものをあらゆる取り寄せ方で取り寄せる場合に対応可能
- ちなみに URI(Uniform Resource Identifier) = URL(Uniform Resource Locator) + URN(Uniform Resource Name)。
- URN は「取り寄せ方」に関係しないような指定方法。例: 「urn:isbn:1-234567-8」とか。

□ Q. リンクをクリックしたとき、元サーバと行き先ページのサーバとの間で通信は行われると思いますか?

□ WWW の原理…

- (1) ブラウザはサーバからコンテンツ (HTML ページ) を取り寄せ表示
- (2) リンク選択→そこに URI 情報が埋め込まれている→ (1) に戻る



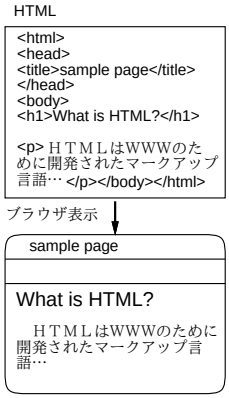
□ つまり全部ブラウザ側だけで処理

□ 実際のブラウザとサーバのやりとり

- URL「http://x.jp//a/b/c.html」を取り寄せるとすると
- (1) ホスト「x.jp」のポート 80 番 (http) に接続
- (2)「GET /a/b/c.html」でファイル取り寄せ
- (3) もし HTML 中に画像ファイル等の参照があれば、それらも同様にして取り寄せる (すべてブラウザ側が主導)
- 利点:非常に簡単なプロトコル→サーバの仕事が単純 (多数をサポート)

□ HTML:非常に簡単なマークアップ言語

- マークアップ:テキストに「印」を埋め込んでおく方式
- 見出し、段落、箇条書き、表、などのタグ
- cf. WYSIWYG:見たまま (Word 等)



□ HTML のサンプル

- これをファイルに入れておく→ブラウザが取り寄せ→ページとして表示

```
<html>
<head>
<title>sample page</title>
</html>
<body>
<h1>What is HTML?</h1>
<p>HTML は WWW のために開発されたマークアップ言語です。</p>
</body></html>
```

□ Q:なぜ Web ページは Word みたいに見たままで作成できないのか?

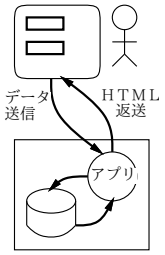
□ 解答:「見たようす」はマシンごとに違うから

- Web ページは世界中のあらゆるマシンで見られる可能性がある
- だから自分のブラウザで見て OK でも完成ではない
- 特にブラウザ独自の機能を使っていると他では見えないことも
- →標準 (HTML であれば WWW コンソーシアムの標準) 準拠が大切

□ Q:Web サーバが FTP(ファイル転送) サーバと最も違うところは何?

□ 解答:「プログラムを動かして出力を返す」機能がある

- 任意のプログラム (CGI) で任意の処理が行える
- ブラウザ画面からデータを渡す機能もある
- ショッピングサイト等もこの原理で作成



□ Perl 言語による CGI(簡単な Unix コマンドを実行し結果を表示)

```
#!/usr/local/bin/perl
print <<EOF;
Content-type: text/html; charset=euc-jp  ← HTTP ヘッダ

<html><title>sample</title></html><body> ←ここから HTML
<h1>w コマンドの実行</h1><pre>
EOF
exec '/usr/bin/w';
print <<EOF
</pre></body></html>
EOF
```

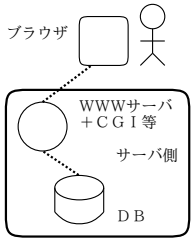
□ フォームによる結果の受け取りとファイル出力

```
#!/usr/local/bin/perl
use CGI; $cgi = new CGI;
print <<EOF;
Content-type: text/html; charset=euc-jp

<html><title>sample bbs</title></html><body>
<h1>書き込みの保存</h1>
<form method="post"><p>
書き込み:<input type="text" name="tx" size="40">
<input type="submit"></p></form></body></html>
EOF
if(defined $cgi->param('tx')) {
  open(FD, ">>tx.data"); print FD $cgi->param('tx'), "\n";
  close(FD);
}
```

□ 実際には「ブラウザ⇔ Web サーバ+CGI 等⇔ DB」の 3 層構造

- データベースが確実なデータ保管を可能にする
- 複数クライアントからの競合アクセスにも対処



□ Q:Web アプリケーションが従来のアプリに比べ革新的な点は?

- 最初から分散アプリケーション
- クライアント側に何もインストール不要
- クライアント側は最初から出来ている
- サーバ側はどんな言語でどういう開発でも OK
- バグ/トラブル対応が全部サーバ上だけで済む

□ クライアント側 (ブラウザ上) でもコードが動かせる

□ JavaScript このために開発されたスクリプト言語

□ Q:どんな利点/用途があると思いますか?

- ブラウザ上→速い応答が必要な場合に使う
- 例:フォーム記入時のエラーチェック、メニュー等ユーザインタフェースの工夫、お遊び、飾り、ゲーム
- Document Object Model(DOM): ページ内容 (文書) を任意に操作

□ JavaScript でページ内容を動かす

```
<html><head>
<title>sample page</title><script type="text/javascript">
var e = null, x, y, vx = 5, vy = 3;
function click(v) {
  e = document.getElementById('e0');
  if(window.event) v = window.event;
  e.style.position = 'absolute'; x = v.clientX; y = v.clientY;
}
function step() {
  if(e == null) return;
  e.style.left = (x+=vx)+'px'; e.style.top = (y+=vy)+'px';
  if(x<0 && vx<0 || x>400 && vx > 0) vx = -vx;
  if(y<0 && vy<0 || y>300 && vy > 0) vy = -vy;
}
</script></head>
<body onload="document.body.onclick=click;setInterval(step,20)
<h1 id="e0">What is HTML?</h1></body></html>
```

□ Q:逆にどんな制約/注意点があると思いますか?

- サーバ上のデータにはアクセスできない (当り前)
- ブラウザ上で実行→セキュリティ上の制約
- しかも、しばしばセキュリティホールに

□ 例:クロスサイトスクリプティング (XSS)

- 掲示板/リンク登録など「書き込める」サイトに対して、
- 「<script>実行したいコード</script>」を書き込む
- それを「見た」人のブラウザ上で…
- 「見た」人が「知らないうちに」…
- 何でもできる (ひどい書き込みをする等)
- ショッピングサイトとかだと勝手に購入とか…

□ 「WWW で次のようなことができるといいなあ」と誰か (たぶん上司) が言ったとします。あなたは次のどれを回答するべきか選びなさい。

- W:「そんなことはもうできていますが…」
- X:「確かに、いずれできるようになったら便利ですね」
- Y:「そんなことは原理的に不可能ですよ」
- Z:「そんなことは絶対にできてくれてはこまります」

□ Q:Web サーバ上でサーバ管理者が用意した任意のプログラムがユーザから入力を受け取って実行できて、結果をユーザに返せるといいね!

□ 解答:W。それはまさに CGI のこと。どこでもやっている。

□ Q:Web サーバ上でユーザが指定した任意のプログラムがユーザから入力を受け取って実行できて、結果をユーザに返せるといいね!

□ 解答:Z。ユーザが好きなプログラムを動かせる→サーバ上のデータでも何でも盗み放題になる→絶対にできてはいけない!

□ Q:Web ページを表示させるとそこに指定されている任意のコマンドが自動的に手元のマシンで実行されてデータ処理してくれるといいね!

□ 解答:Z。たまたま見たページに仕込まれているプログラムが自分のマシンで実行されたら→データ盗まれ放題、破壊され放題→絶対にできてはいけない!

□ Q:Web ページを表示させるとそこに指定されているファイルを自動的に手元のマシンから吸い上げてデータ処理・表示されるといいね!

□ 解答:Z。たまたま見たページに指定されたファイルが勝手にサーバに吸い上げられたら→データ盗まれ放題→絶対にできてはいけない!

□ Q:Web ページを表示させて眺めただけでどういうデータ処理が必要かが読み取られてデータ処理・表示されるといいね!

□ 解答:Y。コンピュータに超能力はありません。

7 まとめ

□ ビジネスインフォマティクス (BI) → 非専門家を準専門家に育成するための系統的なメソッド/カリキュラム

- そのコアアイデア → 「なぜ」を考えて納得する

□ 例:ネットワーク → 「つながる」ための緻密な体系

- 「なぜそうなのか/そうなっているのか」はすべて理由が…

□ 例:WWW → サーバ側/クライアント側プログラミングによる任意の動作→ Web アプリケーション

- 「どういうことはできるか/できては困るか」すべて理由が…

□ 本来はこれらを「自分で確認しながら」身につけて行く必要