4. B2C,B2E: Concepts and Architectures

4.1 Business-to-Consumer Systems

Architectures and Components

Shop Functionalities, Selected Components

4.2 Electronic Fulfillment & Payment

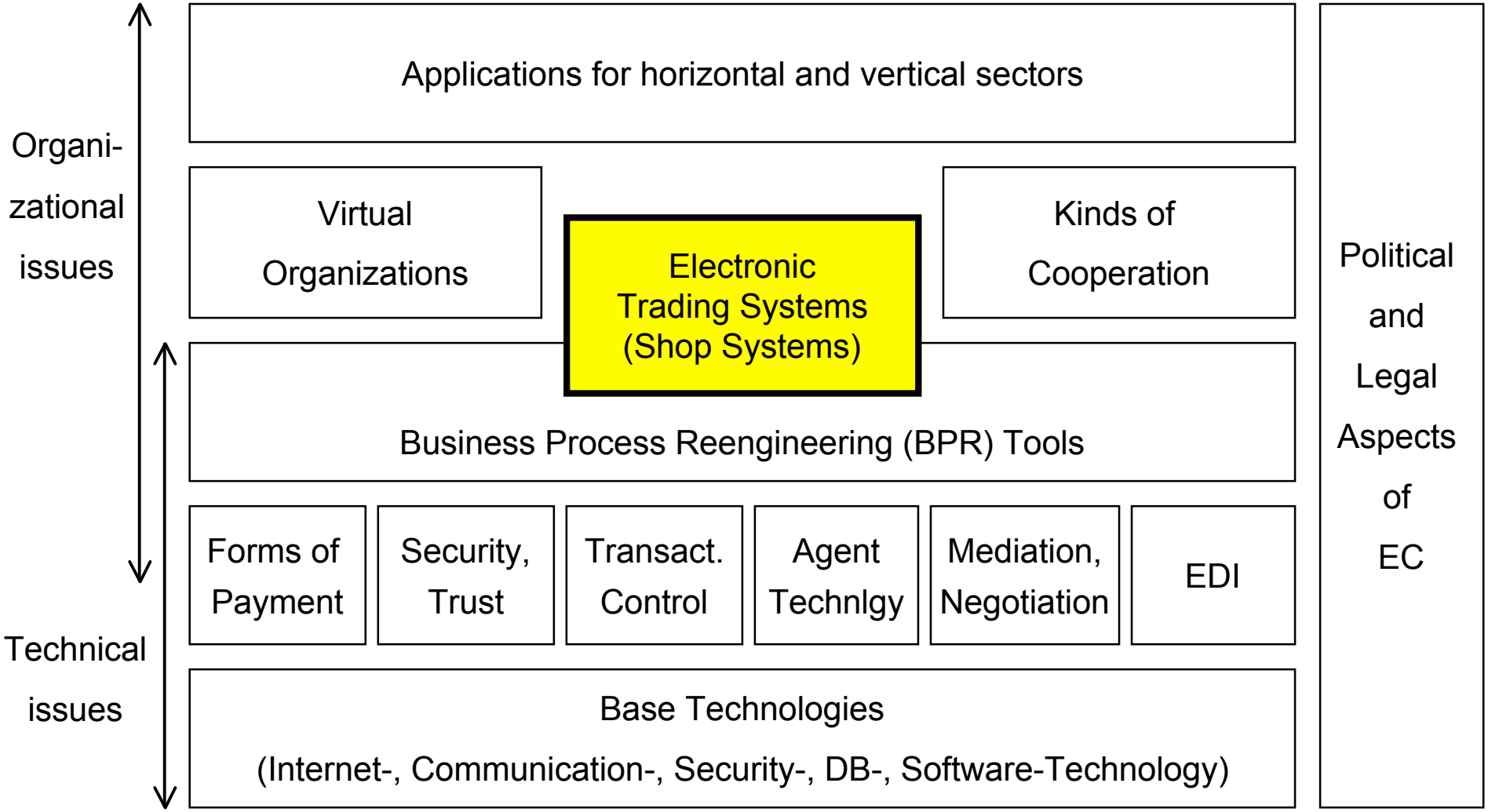
Secure Communication, Security and Trust

Encryption: Standards, Authentication: Digital Signatures, Certification Authorities

Electronic Payment Models, Standards and Systems

4.3 Mobile E-Commerce and Location-Based Services

ECommerce Reference Model



[MeTuLa99]

Online Shops

An **Online Shop System** defines the buyer / seller interface using Internet technology. It supports mainly the Business-to-Consumer (B2C) business model.

Shop system vendors have established knowledge about online shops (e.g., Intershop exists since 1994).

Shop vendors can be classified as

- specialized shop providers (Intershop, OpenShop, ...)
- standard software providers (Microsoft, Oracle, ...) which integrate shops into their software suites.

An online shop is more complex than it seems to be at first glance.

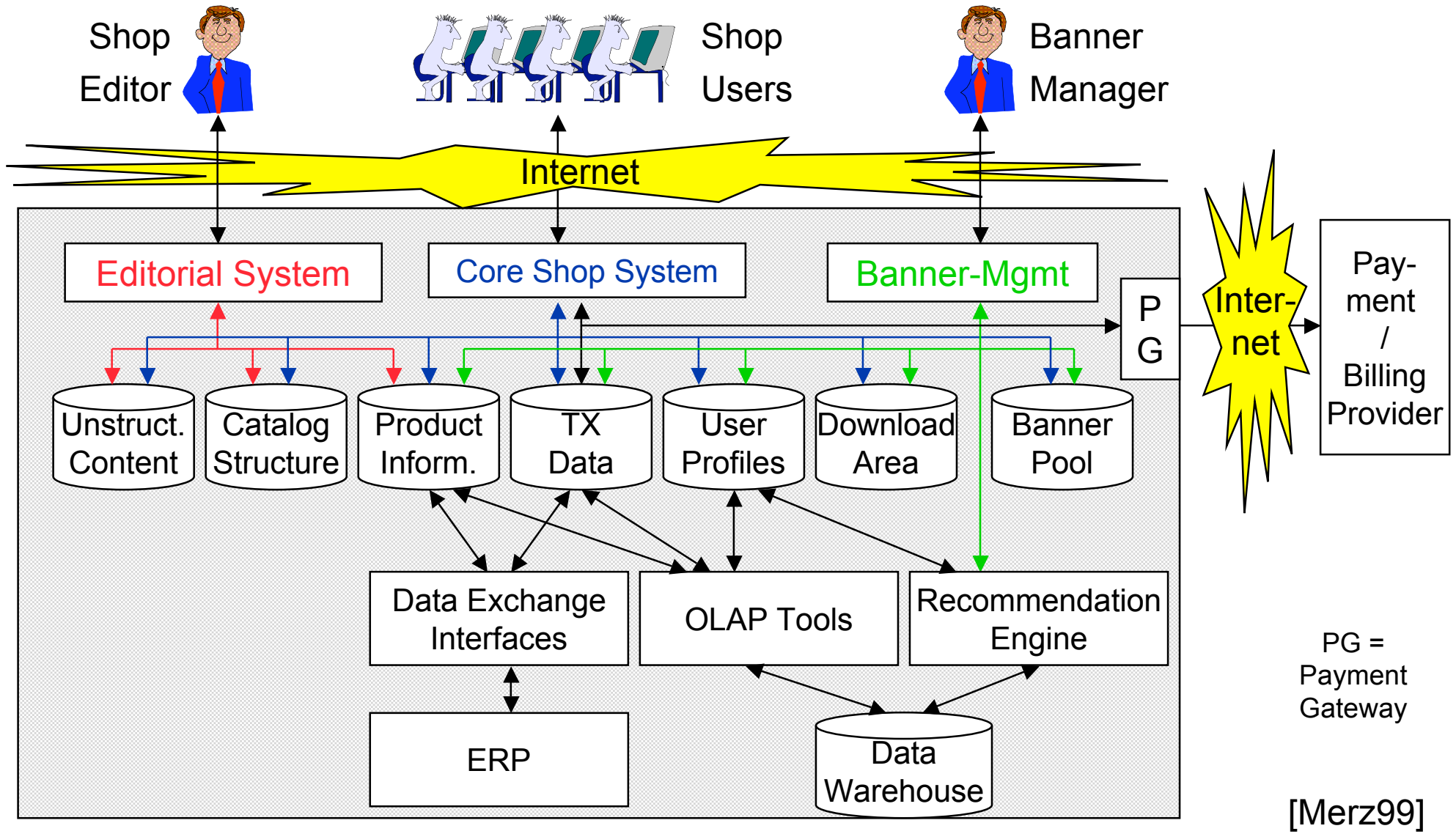
Online Shops: Core Functionality

- **Product Catalog Management** maintains a (flat, hierarchical, …) directory of
  - Hard (tangible) goods (examples: books, cars, clothing)
  - Soft (intangible) goods (example: software product, license, news)
- **Search Engine**
- **Shopping Basket Management** links customers & products
- **Customer Identification** (example: visitor vs. buyer vs. preferred customer, …)
- **Billing & Payment**
  - Taxation
  - Shipping fees

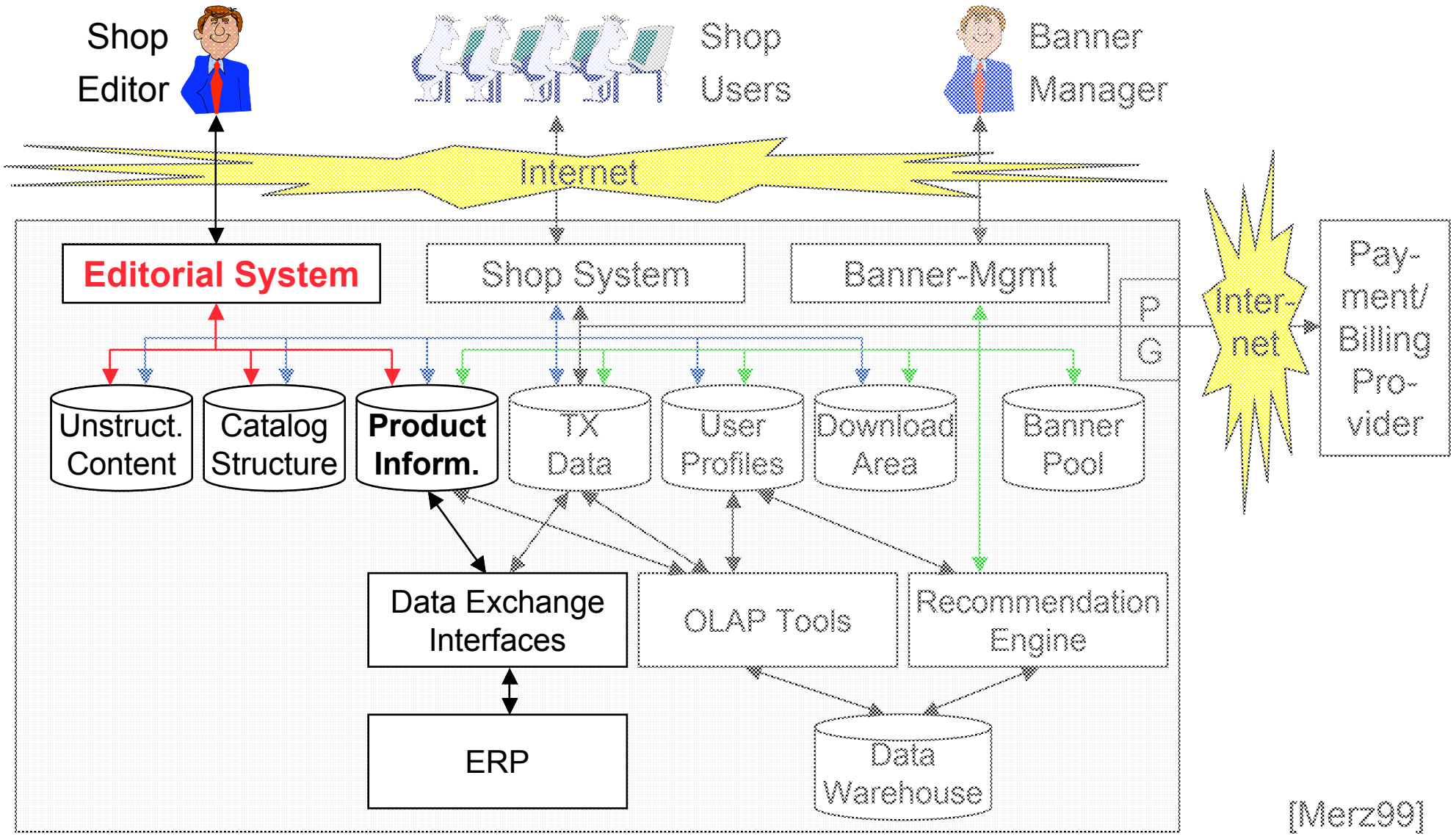
Online Shops: Desirable Functionality

- **Banner Management**
- **Statistics Report, Data Mining**
- **Integration of Customer Relationship Management (CRM) functionality**
  - Customer Profiling
  - Customer Classification
  - Call Center Integration
  - Campaign Management
- **Bridge to Enterprise Resource Planning (ERP) Backend System**
  - Controlling
  - Inventory management (“are there sufficient hard goods on stock?”)
  - Accounts payable (German: *Kreditoren*)
- **Legal issues: Electronic Contracts**
  - Exchange for binding legal statements / electronic contracts between customer & merchant

Online-Shop: Reference Architecture



Selected Shop System Components: Editorial System



Selected Shop System Components: Editorial System

An **Editorial system** is used to manage and administrate a shop:

- Manage
  - product information
  - product catalog structure
  - unstructured content (examples: logos, headline, footer, copyright notice)
- Import product information from ERP systems

NOTE: Sample online shop for starting is usually provided by shop system vendor

Selected Shop System Components: Product DB

The **Product Information Database** contains

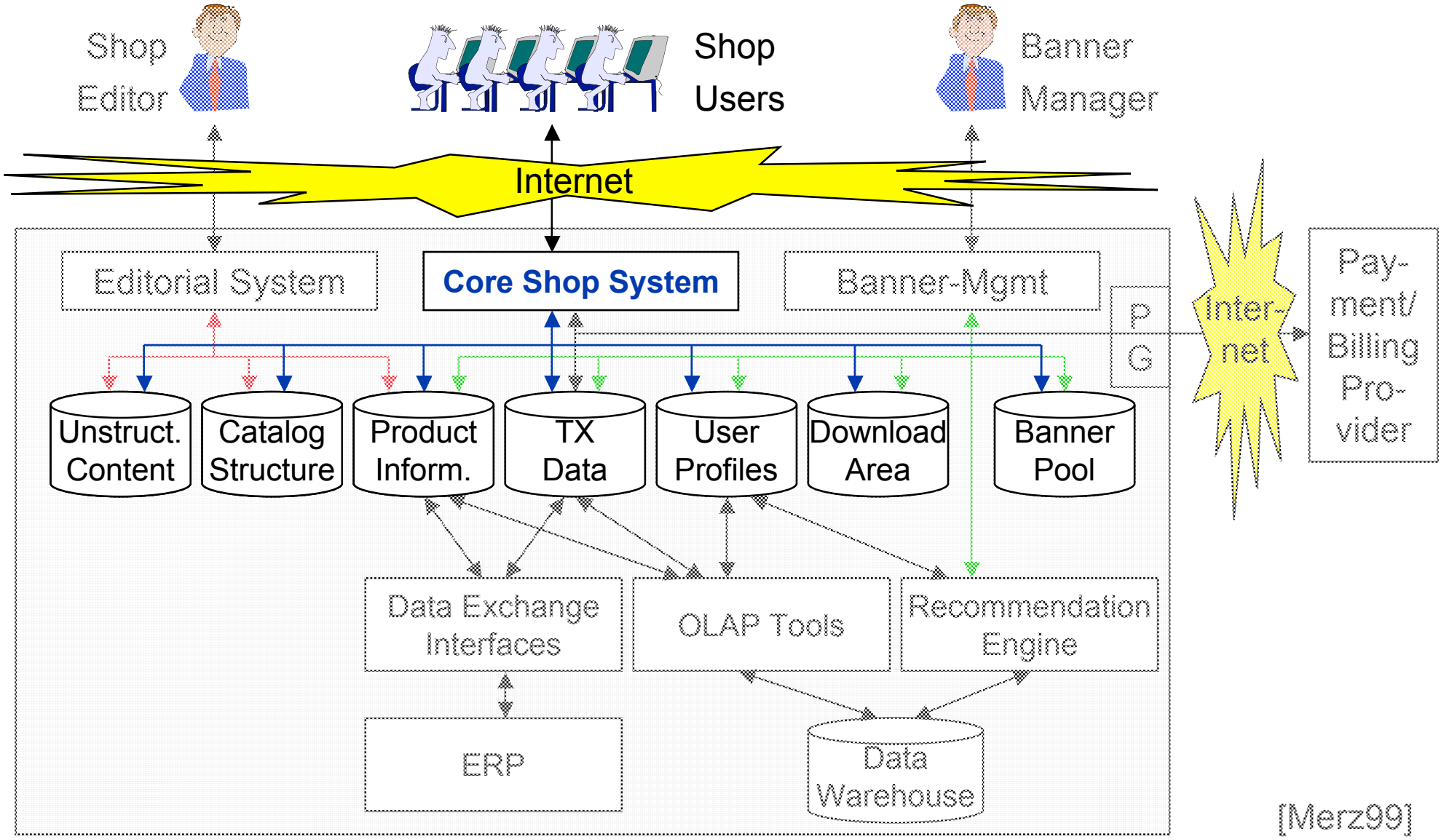
- Product classification information (categorization) and
- Product information, e.g.:
  - Product attributes (name, price in different currencies, ...)
  - Image / 3D model
  - Descriptions
  - Discounts, Advertising intensity
  - Links to related products

Product information databases may support staging process (see later).

Required: Adaptability and extensibility of

- Categorizations
- Product specifications (attributes)

Selected Shop System Components: Core Shop System



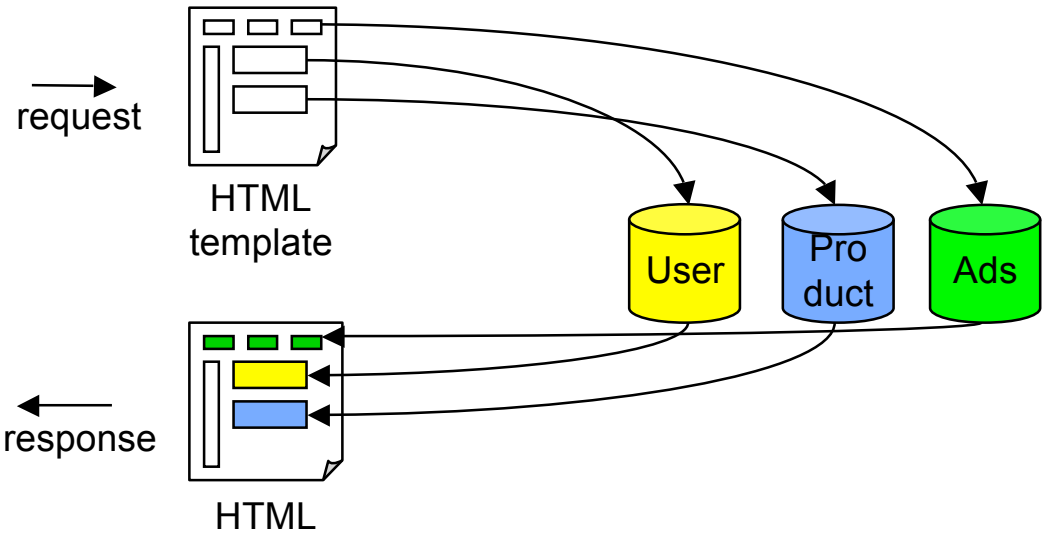
Selected Shop System Components: Core Shop System

A shop software **presentation system** is part of the core shop system.

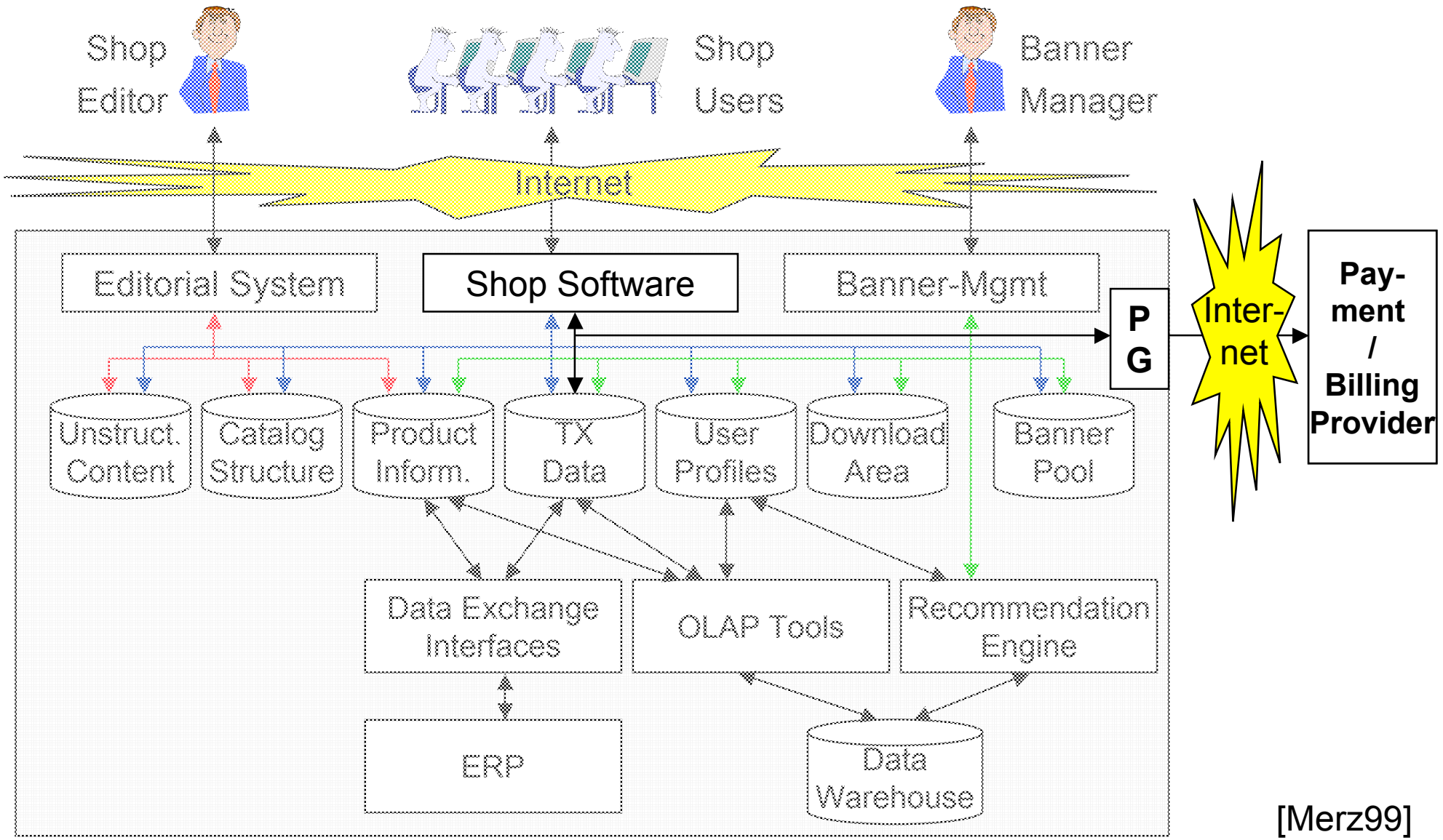
It controls the visualization aspects of an online shop. Two main objectives:

- Aggregation of content to be displayed as a whole. A page contains, e.g.,
  - product information
  - ads from banner pool
  - news feed
  - personalized content
  - download links, ...
- Rendering of content
  - format
  - resolution

See chapter 3 for architecture and realization technologies.



Selected Shop System Components: Payment Gateway



[Merz99]

Selected Shop System Components: Payment Gateway

A **Payment gateway** is the interface from the shop system to a banks‘ clearing server.

- Not developed by shop system providers, but by electronic payment system vendor.
- Separately bought as plug-in / cartridge
- Usually integrated as a shell script or component

Payment Gateway vendors usually provide credit-card payment clearing service only.

4. Shop Systems: Concepts and Architectures

4.1 Business-to-Consumer Systems

Architectures and Components

Shop Functionalities, Architectures, Selected Components

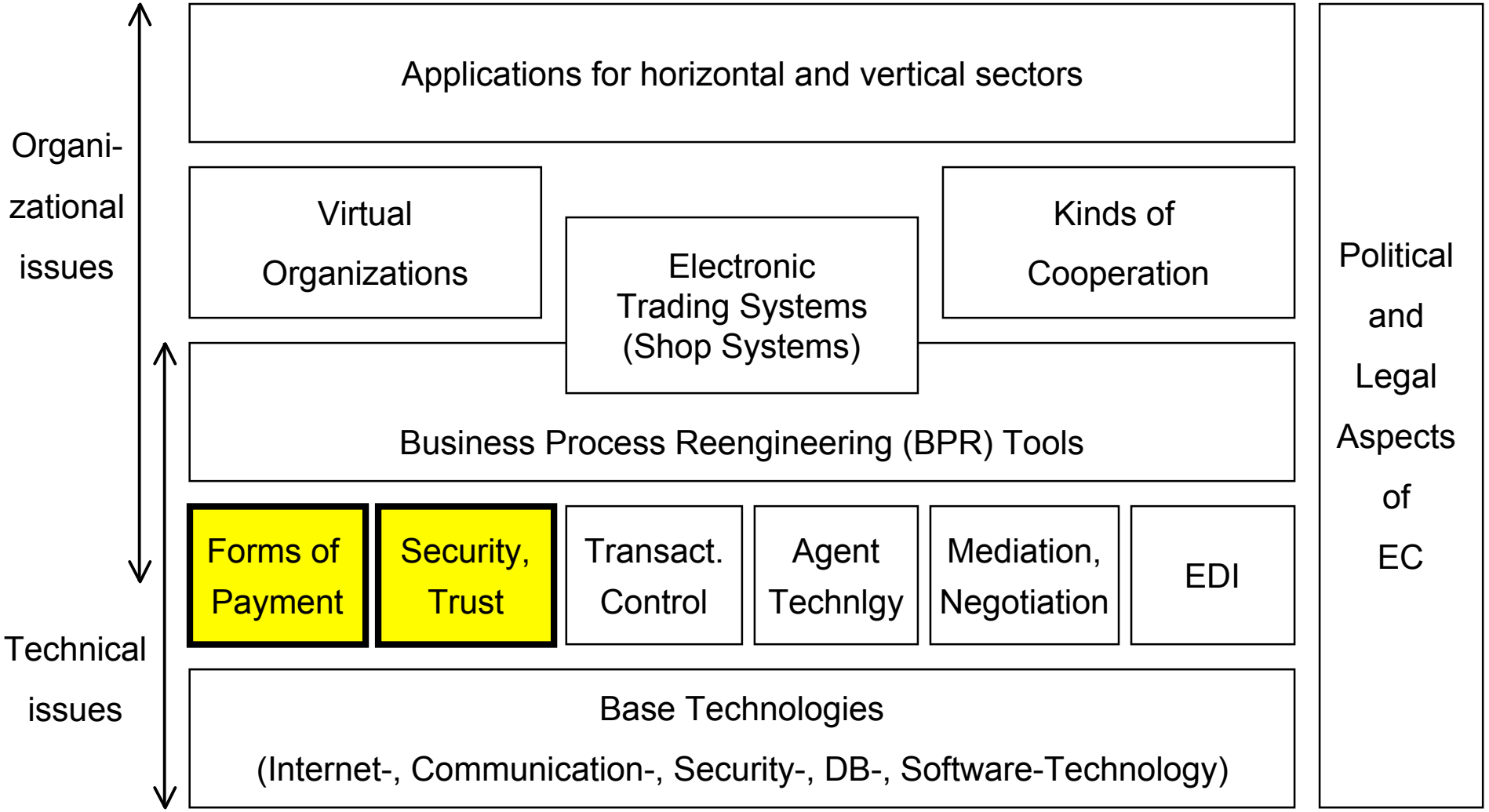
4.2 Electronic Fulfillment & Payment

Secure Communication, Security and Trust

Encryption: Standards, Authentication: Digital Signatures, Certification Authorities

Electronic Payment Models, Standards and Systems

ECommerce Reference Model



[MeTuLa99]

Security and Trust (1)

Security and **trust** are important factors for ECommerce:

- In commerce the *security* of money and goods is most important.
- Buyers will only trade with sellers whom they *trust*.

Security can be achieved technically (e.g., by encryption, authentication, access control to resources).

Trust cannot be achieved technically, can be gained by

- being certified by a trusted organization (e.g., Trust-E)
- number of customers (increases confidence)
- word of mouth (tell your friend)

Security (1)

First step: Secure communication:

Business partners send authentication information, digital cash, contracts over the Internet. For this secure communication is needed. Secure transmission of web documents is achieved, e.g., via Secure HTTP (see chapter 3).

But this is not enough for **secure commerce.** Questions concerning secure commerce:

- Customer view:
  - Ensure that the merchant is who he claims to be (merchant authentication)
  - Ensure merchant is not selling customer profile information (see profiling standards)
- Merchant view: Problems with “interrupted” business processes: Customer denies having
  - placed an order
  - signed a digital contract (**repudiation**).

Security (2)

Problems and proposed solutions (details on following slides):

PROBLEM	**SOLUTION**
Eavesdropping: Third parties (man in the middle) read exchanged documents (orders)	Privacy (Encryption)
Modification of documents (e.g., orders)	Integrity (Digital Signatures), Privacy (Encryption)
Erasure of documents	Backups, Multiple Transmissions, Mirrors
Spoofing (simulate an identity)	Certificates, Certification Authorities, Public Key Infrastructure (PKI)
Repudiation	Non-repudiation (Sign contract with digital signature)

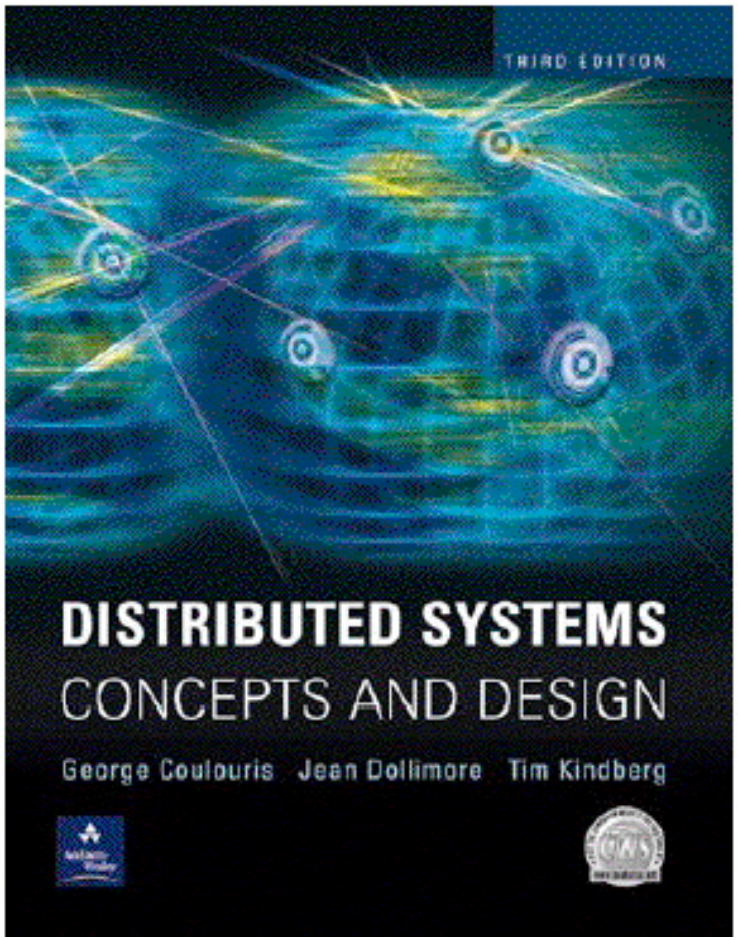
Trade off between effort and risk (when to apply security measures).

Example: Securing documents against eavesdropping / modification:

- Irrelevant for information browsing
- Relevant when credit-card numbers, amounts of money are sent across the Internet

Cryptography in E-Commerce

Based on:



Introduction

Cryptography: encode message data so that it can only be understood by intended recipient.

Romans used it in military communication

Given knowledge of encryption algorithm, brute force attempt: try every possible decoding until valid message is produced.

Computers are good at this!

Modern schemes must be computationally hard to solve to remain secure.

Cryptographic Terminology

Plain text: the message before encoding.

Cipher text: the message after encoding.

Key: information needed to convert from plain text to cipher text (or vice-versa).

Function: the encryption or decryption algorithm used, in conjunction with key, to encode or decode message.

Key distribution service: trusted service which hands out keys.

Encryption

Encrypting data prevents unauthorised access and modification to the data (i.e. prevents eavesdropping and tampering).

If encrypted data can only be decrypted with a matching key, this can be used to prove sender's identity (i.e prevents masquerading).

Likewise, it can be used to ensure that only intended recipients can use the data.

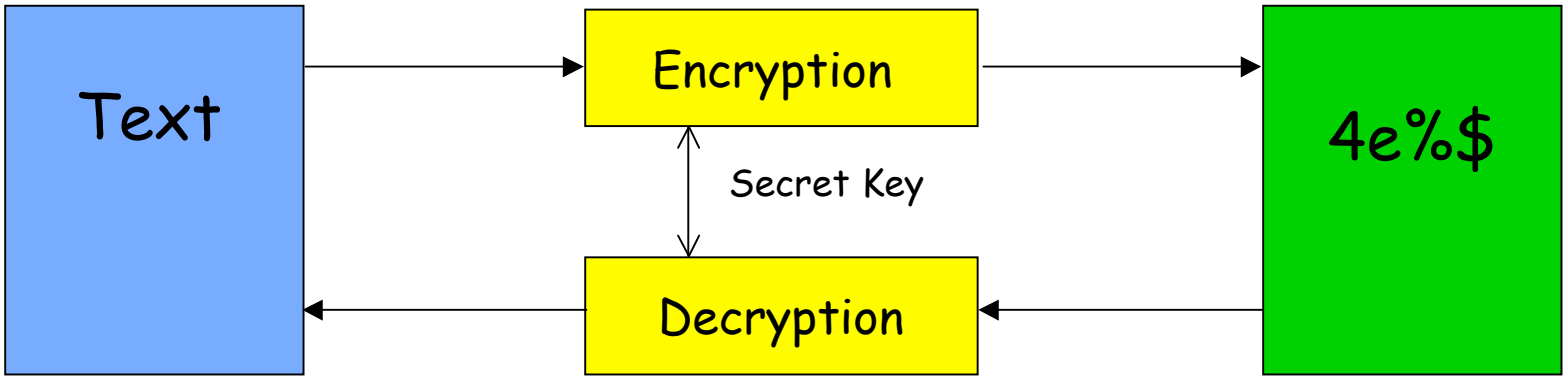
Two main ways: secret key & public key.

Secret Keys

One key is used to both encrypt and decrypt data

Encryption and decryption functions are often chosen to be the same

Security should not be compromised by making function well-known as security comes from secret keys



Using Secret Keys

Sender and recipient exchange keys through some secure, trusted, non-network based means

Sender encodes message using function and sends, knowing that only the holder of key (the intended recipient) can use it

Recipient decodes message and knows that only sender could have generated it

Message can be captured but is of no use

Brute force appraoch for determining K_{AB}

Given Message M‘ = $\{M\}_{KAB}$

For all k

- For all M
  - If M_k= M‘ then return k

Public Keys

Diffie and Hellman 1976

Gives 'one-way' security.

Two keys generated, one used with decryption algorithm (private key) and one with encryption algorithm (public key).

Generation of private key, given public key, is computationally hard.

Do not need secure key transmission mechanism for key distribution.

Using Public Keys

Recipient generates key pair.

Public key is published by trusted service.

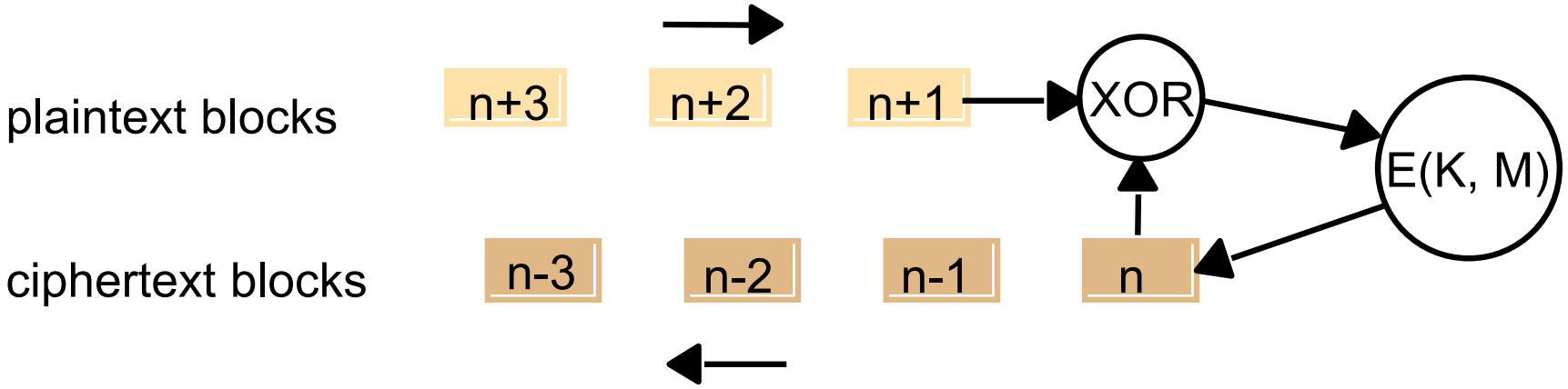
Sender gets public key, and uses this to encode message.

Receiver decodes message.

Replies can be encoded using sender's public key from the trusted distribution service.

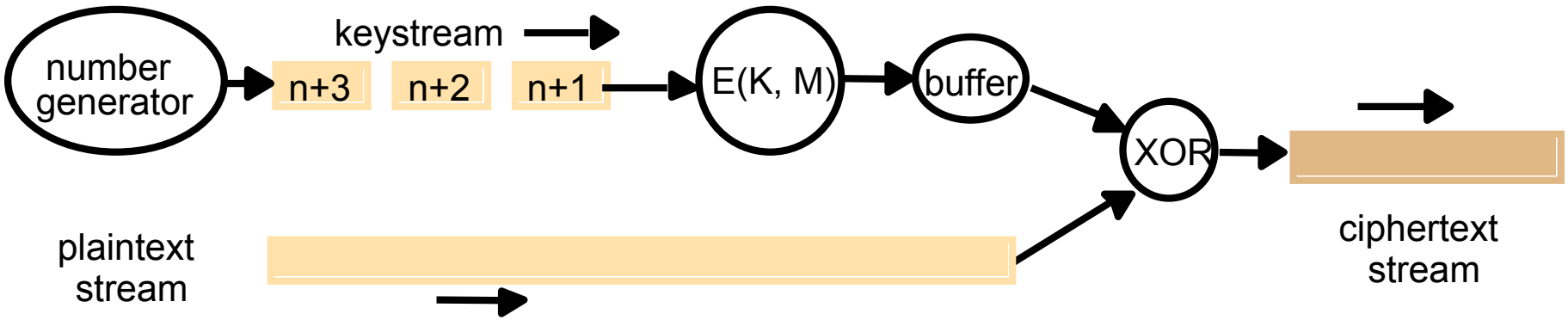
Message can be captured but is of no use.

Cipher Block Chaining



Initialization vector required (e.g., timestamp)

Stream Ciphers



Number generator:
E.g. random number with seed value on which both parties agree

Cryptography: Secret Keys

Main Ideas

- Confusion (XOR, circular shifing, ...)
- Diffusion (transposition of plaintext block portions)

Cyptographic Algorithms

- DES (Data Encryption Standard, 1981, 56bit)
  - Encryption and decryption function identical
- TEA (Tiny Encryption Algorithm, 128bit)
  - Wheeler and Needham 94
- IDEA (Intern. Data Encryption Algorithm, 128bit)
- Blowfish

TEA encryption function

```
void encrypt(unsigned long k[], unsigned long text[]) {
    unsigned long y = text[0], z = text[1];                          1
    unsigned long delta = 0x9e3779b9, sum = 0; int n;                2
    for (n= 0; n < 32; n++) {                                        3
        sum += delta;                                                4
        y += ((z << 4) + k[0]) ^ (z+sum) ^ ((z >> 5) + k[1]);        5
        z += ((y << 4) + k[2]) ^ (y+sum) ^ ((y >> 5) + k[3]);        6
    }
    text[0] = y;  text[1] = z;                                       7
}
```

TEA decryption function

```
void decrypt(unsigned long k[], unsigned long text[]) {
    unsigned long y = text[0], z = text[1];
    unsigned long delta = 0x9e3779b9, sum = delta << 5;  int n;
    for (n= 0; n < 32; n++) {
        z -= ((y << 4) + k[2]) ^ (y + sum) ^ ((y >> 5) + k[3]);
        y -= ((z << 4) + k[0]) ^ (z + sum) ^ ((z >> 5) + k[1]);
        sum -= delta;
    }
    text[0] = y; text[1] = z;
}
```

TEA in use

```
void tea(char mode, FILE *infile, FILE *outfile, unsigned long k[]) {
/* mode is ’e’ for encrypt, ’d’ for decrypt, k[] is the key.*/
    char ch, Text[8]; int i;
    while(!feof(infile)) {
        i = fread(Text, 1, 8, infile);          /* read 8 bytes from infile into Text */
        if (i <= 0) break;
        while (i < 8) { Text[i++] = ' ';}    /* pad last block with spaces */
        switch (mode) {
        case 'e':
            encrypt(k, (unsigned long*) Text); break;
        case 'd':
            decrypt(k, (unsigned long*) Text); break;
        }
        fwrite(Text, 1, 8, outfile);          /* write 8 bytes from Text to outfile */
    }
}
```

Cryptography: Public Keys

$D(K_d(E(K_e,M)) = M$

Decryption key K_d must be a secret

Encryption key K_e is public

RSA Encryption - 1

To find a key pair e, d:

1. Choose two large prime numbers, P and Q (each greater than 10100), and form:
   $N = P \times Q$
   $Z = (P-1) \times (Q-1)$
2. For d choose any number that is relatively prime with Z (that is, such that d has no common factors with Z).
   We illustrate the computations involved using small integer values for P and Q:
   $P = 13$, $Q = 17$ –> $N = 221$, $Z = 192$
   $d = 5$
3. To find e solve the equation:
   $e \times d = 1 \bmod Z$

That is, $e \times d$ is the smallest element divisible by d in the series $Z+1$, $2Z+1$, $3Z+1$, ...

$e \times d = 1 \bmod 192 = 1, 193, 385, ...$
385 is divisible by d
e = 385/5 = 77

RSA Encryption - 2

To encrypt text using the RSA method, the plaintext is divided into equal blocks of length k bits where $2^k < N$ (that is, such that the numerical value of a block is always less than N; in practical applications, k is usually in the range 512 to 1024).

k = 7, since $2^7 = 128$

The function for encrypting a single block of plaintext M is:

$E'(e,N,M) = M^e \bmod N$

for a message M, the ciphertext is $M^{77} \bmod 221$

The function for decrypting a block of encrypted text c to produce the original plaintext block is:

$D'(d,N,c) = c^d \bmod N$

Rivest, Shamir and Adelman prove*d* that E' and D' are mutual inverses (that is, $E'(D'(x)) = D'(E'(x)) = x$) for all values of P in the range $0 \le P \le N$.

The two parameters e,N can be regarded as a key for the encryption function, and similarly d,N represent a key for the decryption function.

So we can write K_e = <e,N> and K_d = <d,N>, and we get the encryption function:

$E(K_e, M) = \{M\}_K$ (the notation here indicating that the encrypted message can be decrypted only by the holder of the private key K_d) and $D(K_d, \{M\}_K) = M$.

Blowfish

Symmetric block cipher encryption/decryption algorithm

Can be used as a drop-in replacement for DES or IDEA.

Takes a variable-length key, from 32 bits to 448 bits, ...

... making it ideal for both domestic and exportable use.

Blowfish was designed in 1993 by Bruce Schneier as a fast, free alternative to existing encryption algorithms. Since then it has been analyzed considerably, and it is slowly gaining acceptance as a strong encryption algorithm. Blowfish is unpatented and license-free, and is available free for all uses.