EXI について

2009年5月12日

米国富士通研究所
上谷　卓己

「*EXI*」って何？

- “**E**fficient **X**ML **I**nterchange” の頭文字
  読み方 は「**エクシィ**」
- XMLデータの効率的な交換形式
- W3C の標準化活動、規格名
  - XMLアクティビティ / EXI ワーキンググループ
  - EXI 規格のラストコール・ワーキングドラフト仕様が公開済。

EXI の背景 – XMLの成功

- “XML” という現象

  ベンダ・サポート、注目・人気、
  エバンジェリスト、本・記事など

- XMLエバンジェリストによる啓蒙活動

  「まだ XML を使ってない？ 時代に乗り遅れるよ」
  「とにかく使ってみてよ。あなたも、きっと気に入るよ」

取り残された者はいなかったのか？

イラストの著作権について
フェアユースであるか判断が
難しいため配布版
からは削除しました。

ご覧になる場合は、下のイラスト名を
クリックしてください。

Great Moments in Evolution

*XML*への移行 – 期待と達成感、失望

- 問題なくXMLを使ったシステムに移行
- 移行してみたがシステム性能が10倍遅くなった…

- 現時点では、性能的に無理。そのうちきっと…

「ぜひ導入したい。XMLの性能問題が解決する日が待ち遠しい」

「少し待ってみて、また再度評価してみたい。」

「ムーアの法則にしたがって、自然と解決するする日を待ちます」

↑

XMLに取り残されたユースケース

代替 *XML* フォーマットのさきがけ

- WBXML - W3C Note、WAP (1999)
  - “XML” と代替可能で、XMLよりも効率よく取り扱い可能な非テキスト・フォーマット
  - 名前空間の扱いが困難。拡張が困難。
  - 特定のワイアレス・ユースケースでのみ利用
  - 多くのユースケースで、効率性が不十分

W3C の消極的関与 (～2003)

- WBXML Note (1999) の事実上失敗
  - 沈黙、自由放任
  - 「技術革新は標準化団体の仕事ではない」と自覚、居直り
- XML Information Set (Infoset) - 2001年勧告
  - XMLに含まれる情報を整理・定義
  - “含まれる”が、含まれない情報を明確化
  - 意図せず、XMLを “XML Information Set” を表す1つのエンコーディングとして考える“見方”を産む

代替 XMLフォーマットの乱立・アナーキズム (～2003)

- W3C による放任。技術の必要性
- 新技術の同時多発的発生

Fast Infoset (Sun)	**Efficient XML (AgileDelta)**
X.694 (ITU-T)	**XEUS (KDDI)**
MPEG-7/BiM (MPEG)	**XSBC (X3D Consortium)**
Xebu (ヘルシンキ大)	**FXDI (富士通)**

⋮ **他多数** ⋮

事態を憂慮したW3C

問題意識

- ドメイン毎に代替フォーマット。フラグメント化
- XMLとの互換性。汎用性

アクション

- XBC WG – XML Binary Characterization WG
  標準化の可能性、必要性を検討
  (2004年3月。2005年3月完了)
- EXI WG 設立 (2006年1月)

EXI の存在を支える考え方

- 「ムーアの法則」に対する疑念
  - XMLの性能問題を解決しないのではないか？
  - 取り残されたユースケースを合法的に見捨てる口実として使われていないか？
- 文書を交換して処理する、というシナリオでは、ドキュメントサイズは性能を左右する主要因。

ムーアの法則 – CPU 処理能力の軌跡

- トランジスタの集積密度が、24ヶ月毎に倍増するという経験則。（CPU処理能力の目安）

CPU	トランジスタ数	リリース年
Intel 80386	275,000	1985
Intel 80486	1,200,000	1989
Pentium	3,100,000	1993
Pentium II	7,500,000	1997
Pentium 4	42,000,000	2000
Itanium 2	220,000,000	2003
Core 2 Quad	582,000,000	2006

- 演算速度の向上で、高速なXML処理が可能
- CPU以外の計算機資源は？

ネットワークという資源の特徴

- ネットワーク固有の特徴
  - ネットワークは共有資源
  - エンドポイント間の最も細い回線で性能が決まる ➔ 性能がネットワークバウンドに陥りやすい
- CPU 資源との関係
  - ムーアの法則によるデータ処理能力の向上
    ➔ より多くのデータを交換する動機付け。
    ネットワーク負荷増大
  - PC・CPUの低価格化。CPUは容易に追加可能
  - ネットワークのスケーラビリティ確保は難しい

CPU に比べてネットワークは希少。

データ交換のドキュメントサイズへの配慮が大切

バッテリの容量

- 計算機の動作には電力が必要
  - モバイルデバイスでは、バッテリ容量・消費量が重要
  - バッテリ容量の進化は、きわめてゆっくり。
- ネットワークとの関係
  - ワイアレス・ネットワークの消費電力は大きい

携帯電話で 1 バイトの送受信で消費される電力
= CPU の処理 130,000 ～ 350,000 サイクルに相当

Nokia 7610 / 123 MHz ARM / GSM Network
(Helsinki Institute for Information Technology)

データ通信のドキュメントサイズは、
携帯機器バッテリ消費量に大きく影響

EXI フォーマット

基本原理

対象についての、より多くの知識は、より効率的な符号化を可能にする。

符号化の対象について、前もって何を知っているのか？

XML文書は、単なる文字の並びか？ バイト列か？

- XMLには規則性がある (XML仕様, XML Infoset)
- XMLスキーマは、より具体的な規則を与える。

EXI の符号化は、この2つの規則を活用する。

EXI の基本文法

- スキーマ定義のない要素は、“**基本文法**”で処理
- 基本文法は、XMLについての**先天的知識**に基づく。

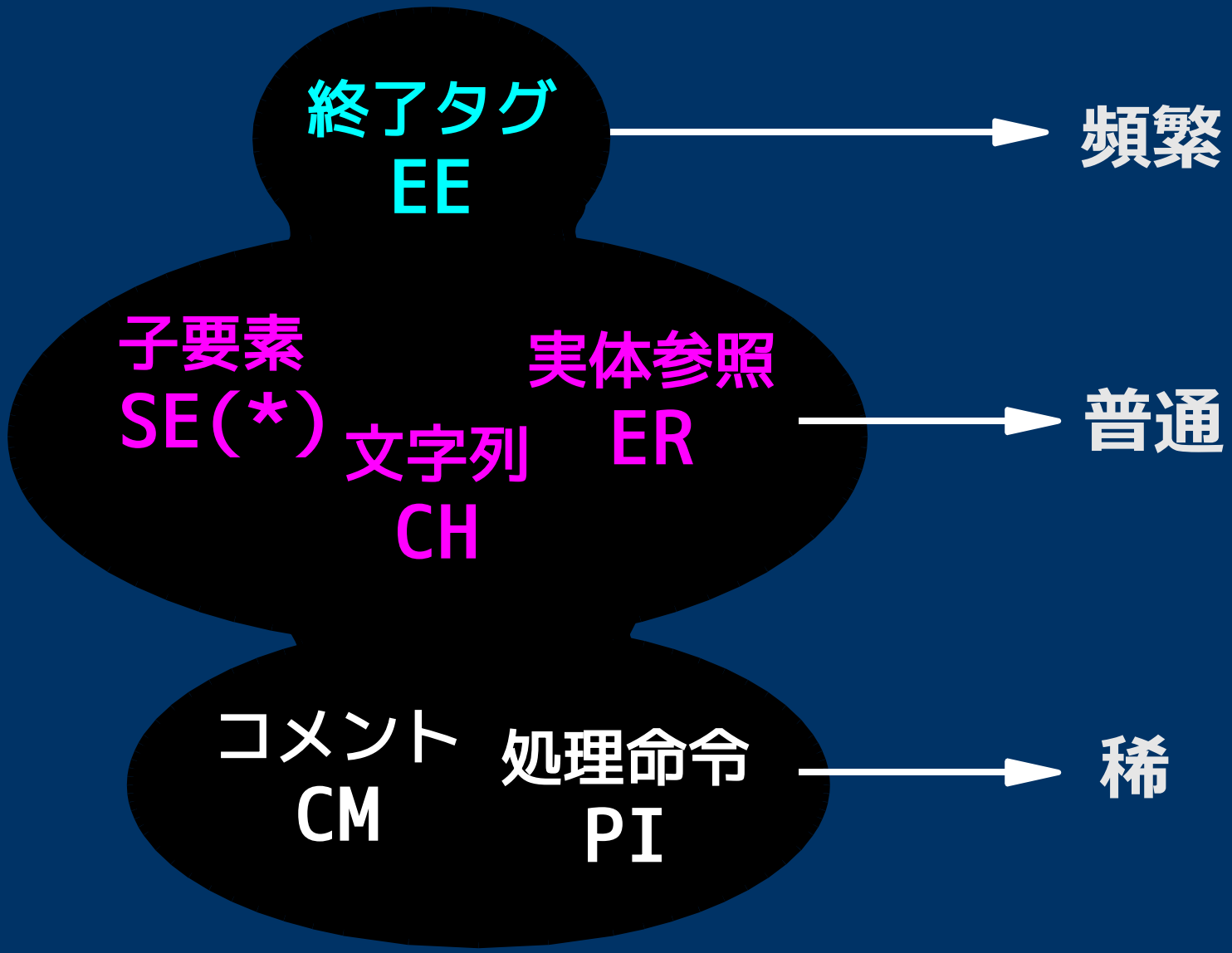
  XMLの先天的知識 ＝ XML仕様書

要素内容として出現可能なもの

<A>

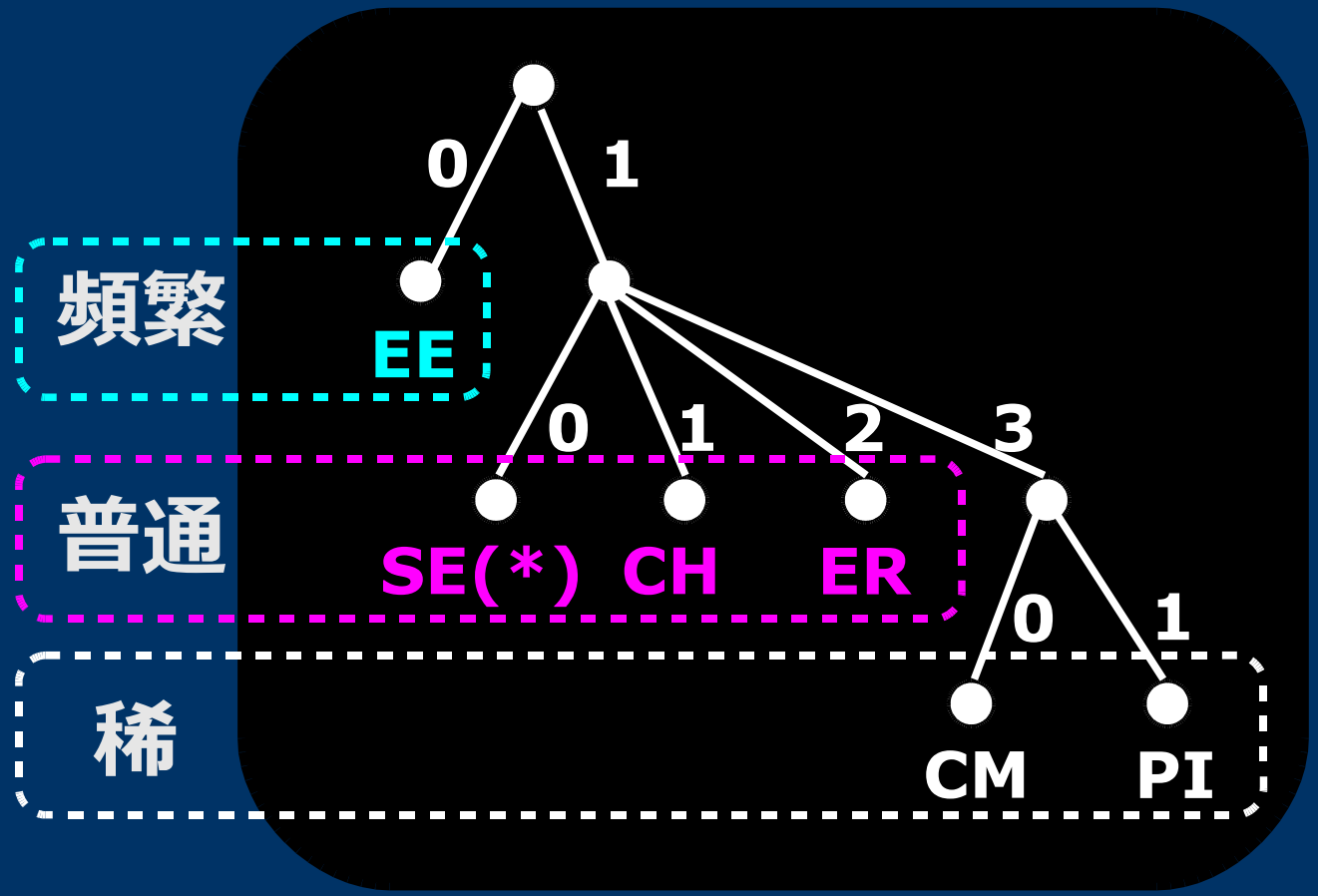
終了タグ	EE
子要素	SE(*)
文字列	CH
実体参照	ER
コメント	CM
処理命令	PI

EXI の基本文法 - 蓋然性による分類



基本文法のイベントコード木

蓋然性に基いて、イベントをツリー（木）に配置



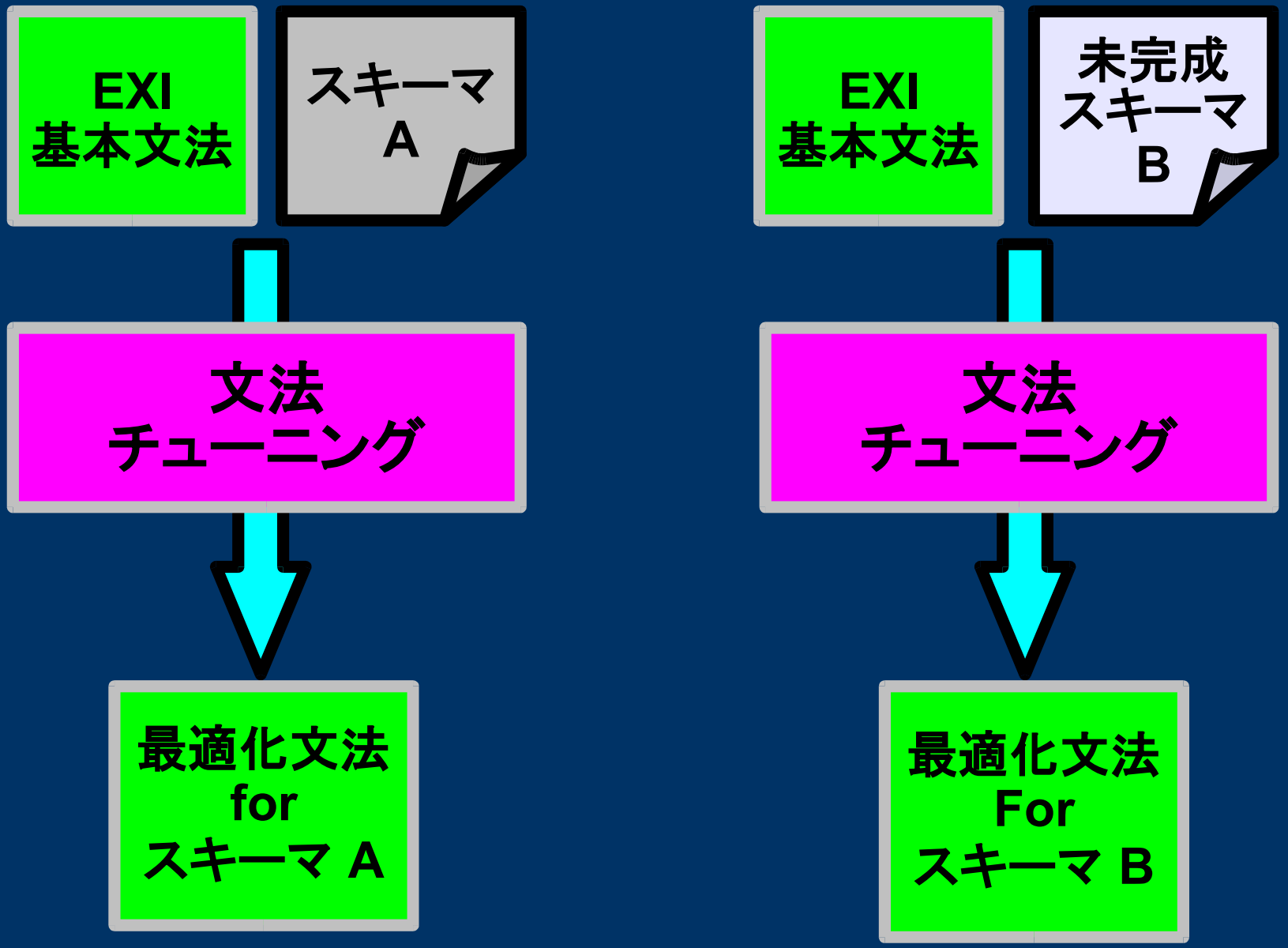
	パス	物理表現
EE	0	0
SE(*)	1.0	1 00
CH	1.1	1 01
ER	1.2	1 10
CM	1.3.0	1 11 0
PI	1.3.1	1 11 1

頻繁　１ビット

普通　３ビット

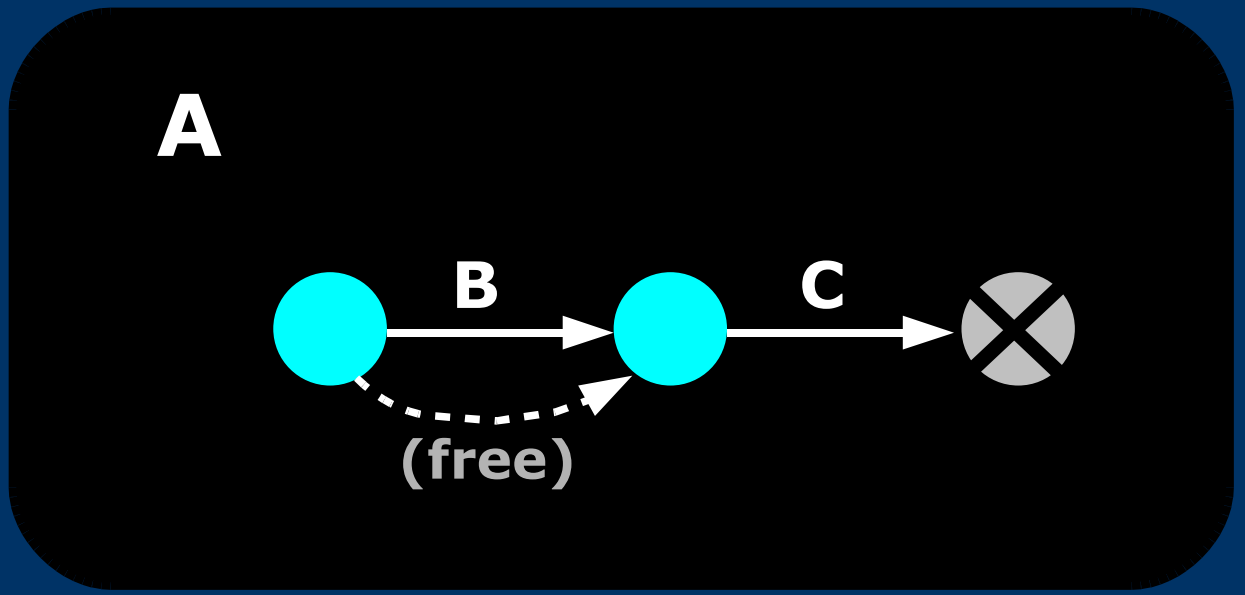
稀　　４ビット

スキーマ・インフォームド文法

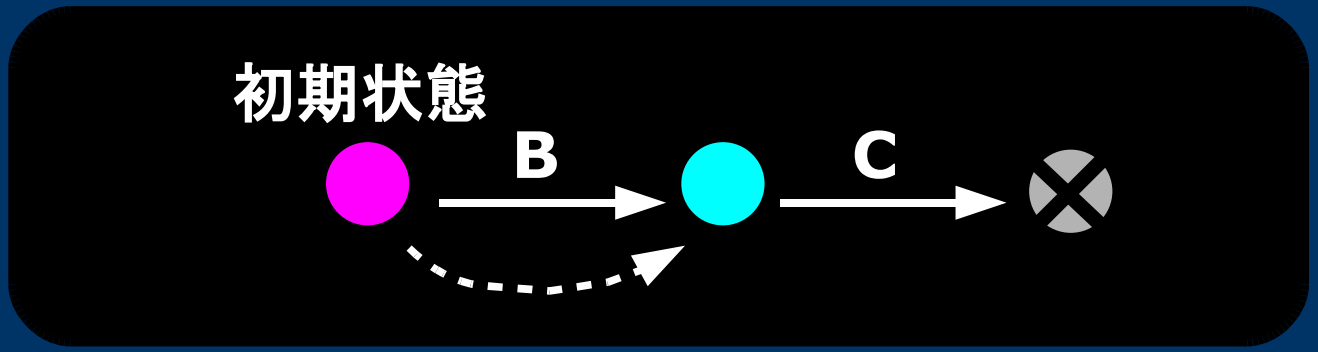


スキーマ・インフォームド文法

A := sequence (B? , C)



スキーマ・インフォームド文法

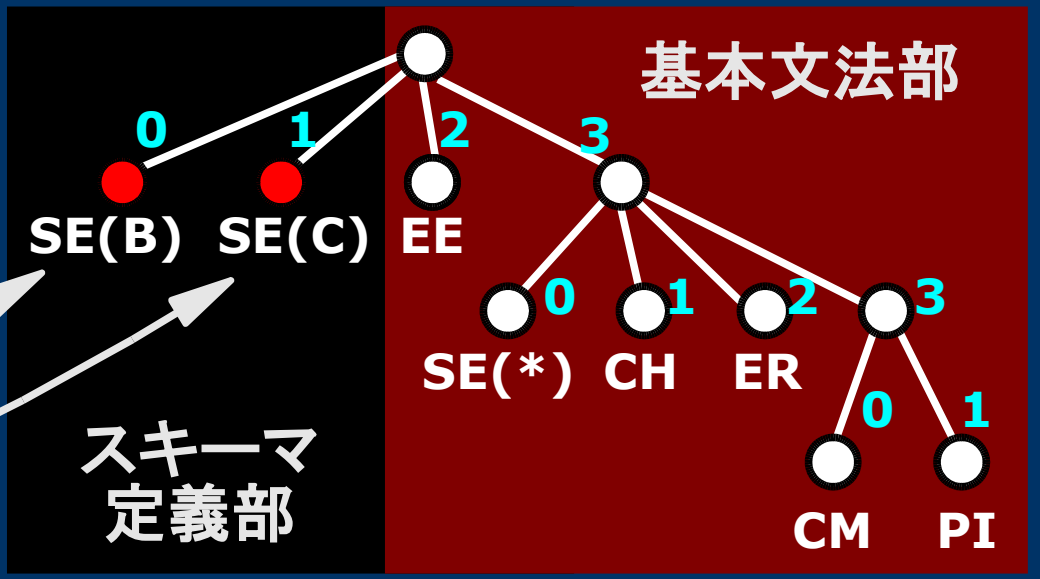


初期状態で出現可能なもの

B
(free) C

B
C

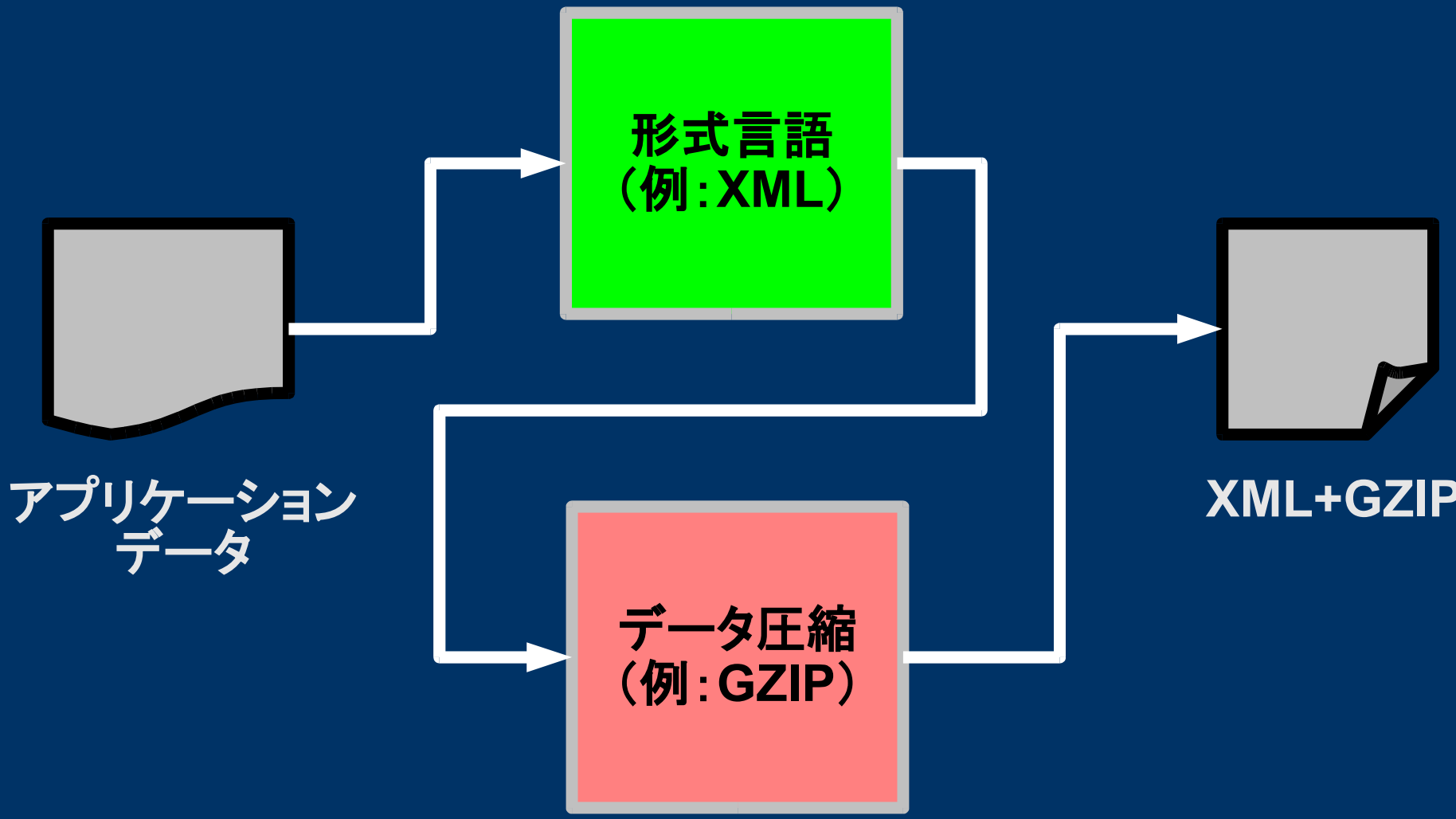
EC1のイベントコード木



スキーマ・インフォームド文法

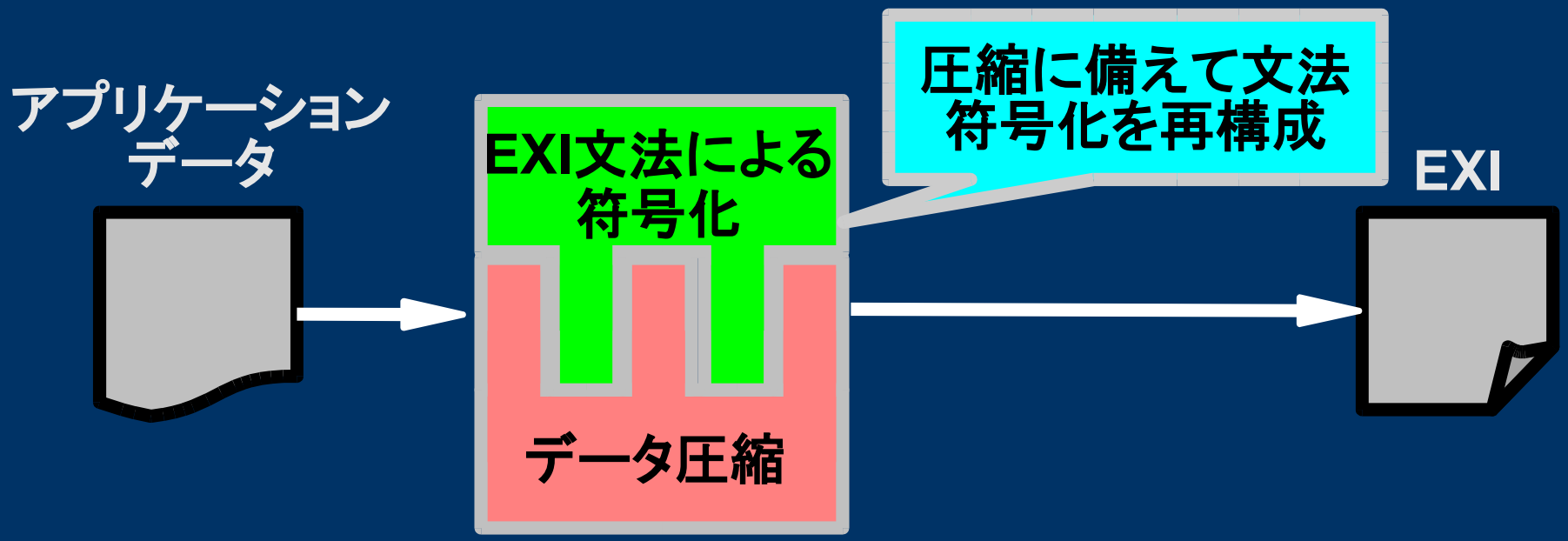
EC1:	パス	物理表現	ビット数
SE(B)	0	00	2
SE(C)	1	01	2
EE	2	10	2
SE(*)	3.0	11 00	4
CH	3.1	11 01	4
ER	3.2	11 10	4
CM	3.3.0	11 11 0	5
PI	3.3.1	11 11 1	5

文法的符号化とデータ圧縮（*XML*）

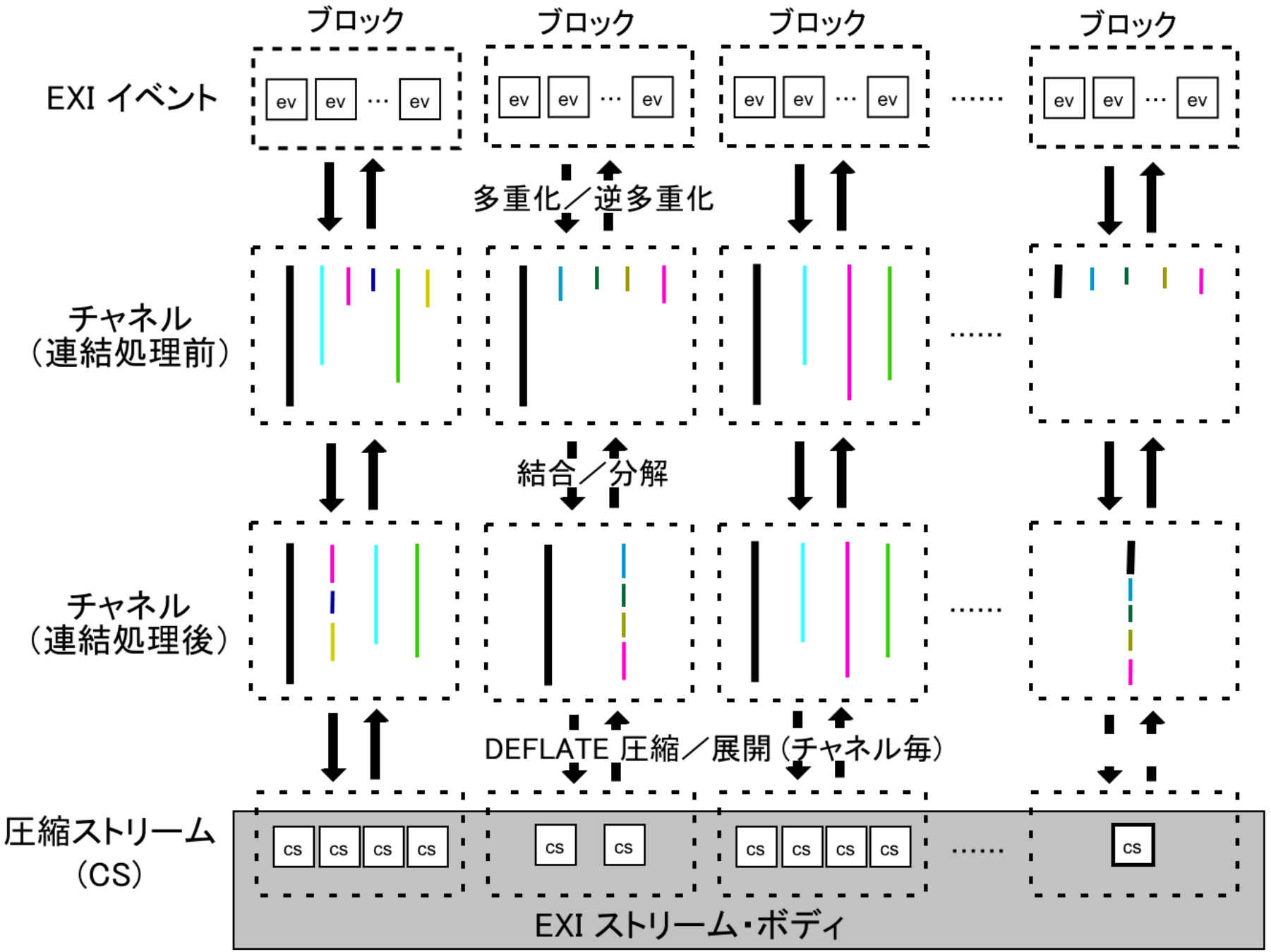


EXI の圧縮機能

- EXI 圧縮は、EXIフォーマットに内在
- EXIの文法システムと DEFLATE を統合
- 圧縮有無は選択可能
- GZIP よりも高い圧縮率
- CPU負荷は GZIP の 1/2 ～ 1/3 程度



EXI 圧縮の仕組み


ブロック
ブロック
ブロック
ブロック
EXI イベント
ev
多重化／逆多重化
チャネル
（連結処理前）
結合／分解
チャネル
（連結処理後）
DEFLATE 圧縮／展開（チャネル毎）
圧縮ストリーム
（CS）
cs
EXI ストリーム・ボディ

EXI 圧縮の効果

	FPML	JTLM
XML	3815 bytes	937005 bytes
XML+GZIP	1292 bytes	113904 bytes
EXI （圧縮あり,スキーマ使用）	345 bytes	7885 bytes
GZIP ／ EXI	3.7倍	14.4倍

FPML: Financial Product Markup Language (金融)
JTML: Joint Theater Logistics Management (軍事)

性能について

EXI のコンパクト性

	圧縮無し	EXI圧縮
スキーマなし	“一貫して” XMLより格段に コンパクト	“一貫して” XML+GZIPと比べて 格段にコンパクト
スキーマあり	“一貫して” ASN.1 PER を 凌ぐコンパクト性	

処理性能

	デコード	エンコード
圧縮なし	14.5倍	6.0倍
圧縮あり	9.2倍	5.4倍

EXI（圧縮ＯＦＦ）対 XML

EXI（圧縮ＯＮ）対 XML+ GZIP

これらの値に、あまり大きな意味はない。

- メモリ経由でローカルファイルを読み込み。
  現実の条件から乖離。
- 処理が CPUバウンドな条件で“XML(+GZIP)より高速であること”を確認。
- 処理性能は実装依存。値は、フォーマットの特性ではない。
- EXI が **高速実装を妨げるフォーマットでないこと** を確認

より現実的な性能分析

エンコード、デコード（プロセッサ処理）は
データ交換と同時に実行される

考慮すべきパラメタ

- 帯域幅（ネットワークの速度）
- 圧縮の効果（帯域幅向上に近い効果）

より現実的な性能値（例）

DVB (Digital Video Broadcasting) CBMS データ
(XML サイズ 428 Bytes)

	11 mbps	54 mbps	loopback
EXI	**6660 TPS**	**15448 TPS**	**84711 TPS**
XML	1060 TPS	**3722 TPS**	**5055 TPS**
XML+GZIP	**1680 TPS**	3559 TPS	3963 TPS
性能比	**4.0**	**4.2**	**16.8**

- GZIP は低速ネットワーク(11 mbps)でのみ有効。
  それ以上のネットワークでは、GZIP の効果なし。
- 無線LAN (11・54 mbps)ではネットワークバウンド
  高速LAN ではCPU(処理性能)バウンド
- すべての帯域幅で格段に高い性能を発揮

データ交換最適化とデータ処理最適化

工場 ＝ アプリケーション

木箱コンテナ

流通＝データ交換

金属コンテナ

EXI の問題領域

- パース
- バリデーション
- 変換
- データバインディング
- アプリ固有処理

XML と EXI

XML と EXI – 新たな二項対立か？

胸のすくステレオタイプ化

- EXI 批判の実例

  - EXI は単なる新種のバイナリ XML
  - バイナリはテキストではない。名前に「XML」を使うな！

  拒絶と不安

- XML批判(仮定)..... (自己矛盾)

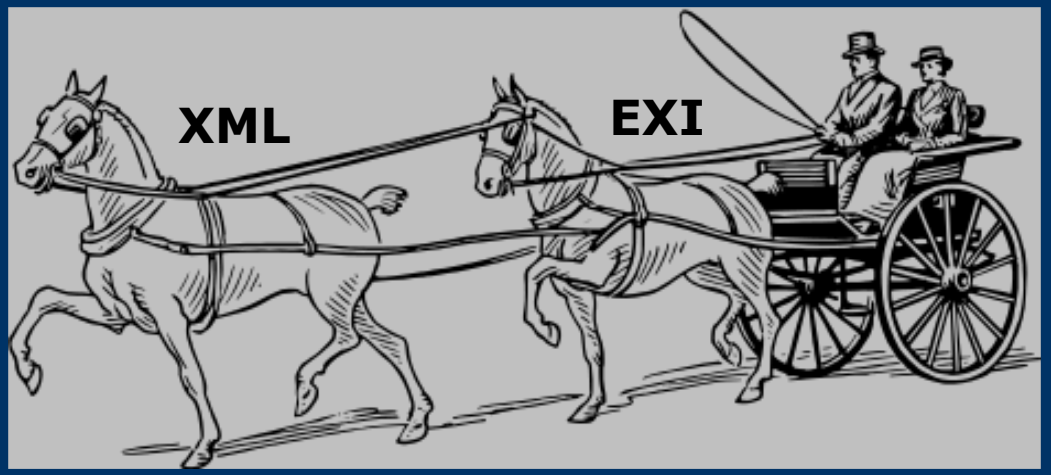
  - XMLは非効率すぎ！

  反抗心

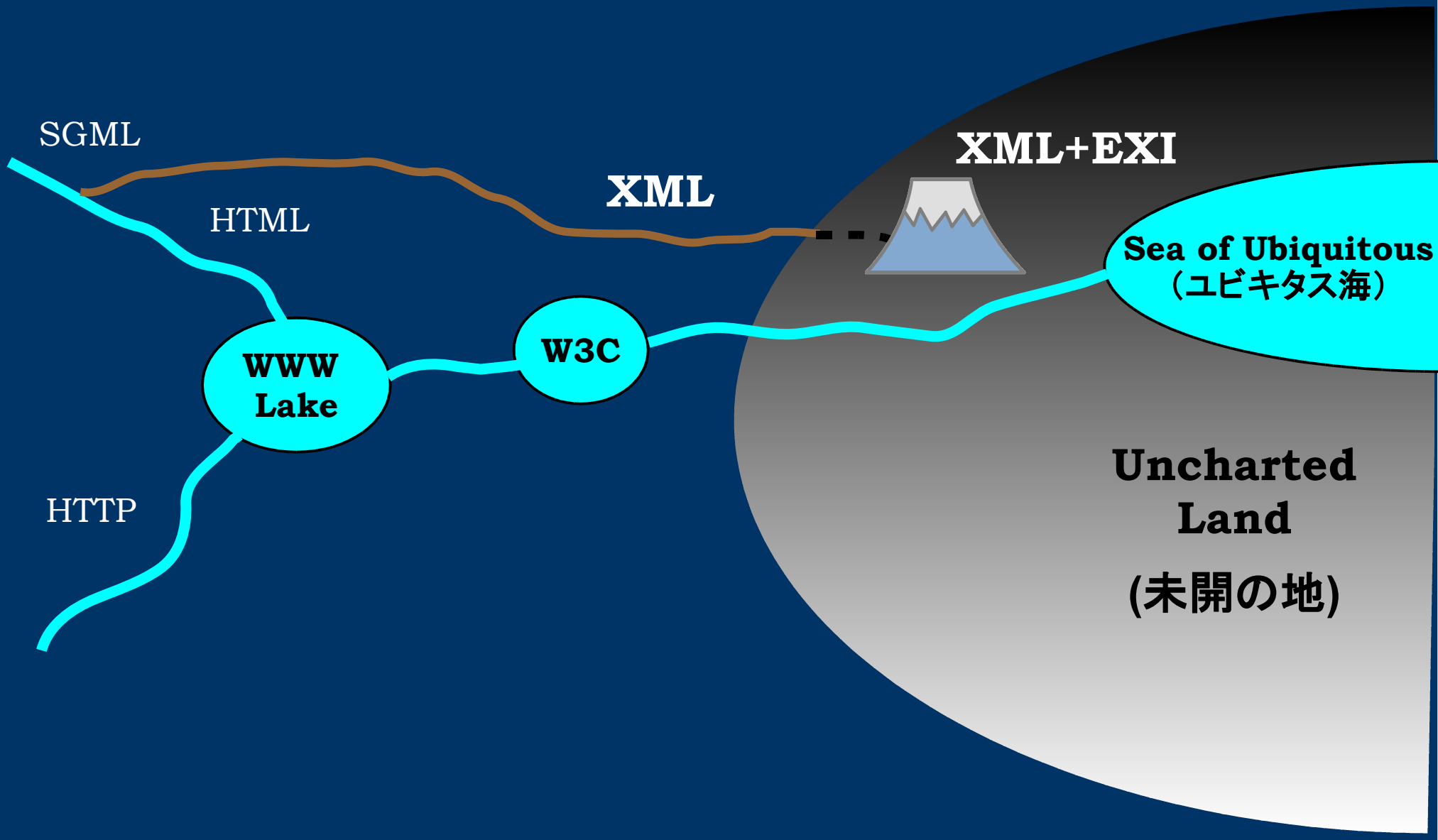
XMLとEXI - 関係の適切な見方

「Efficient XML Interchange」フォーマット

- XMLは EXI の大前提
- XMLの肯定が出発点
- XMLの否定、XMLに取って替わることではない
- **XMLを補うことが大目標**



パイオニア／開拓者 - 募集中



トライアル・評価用のリソース

- EXI ライブラリ（ドラフト仕様の実装）
  - AgileDelta Efficient XML (製品。評価版あり)
  - EXIficient (GPLライセンス / バージョン 0.3)
- 性能評価ドキュメント
  - EXI Evaluation (W3C Note)

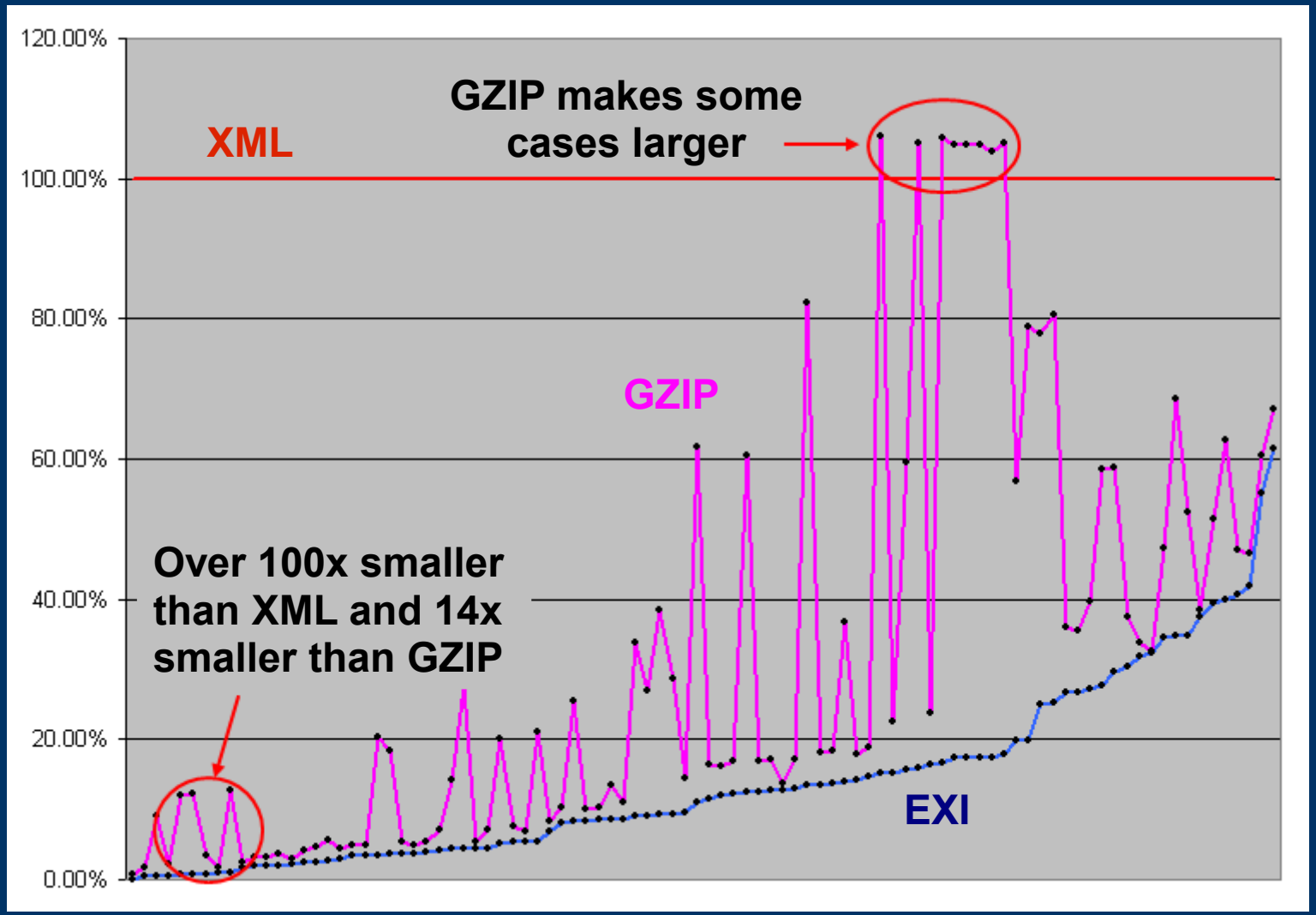
EXI WG ホームページ

http://www.w3.org/XML/EXI/

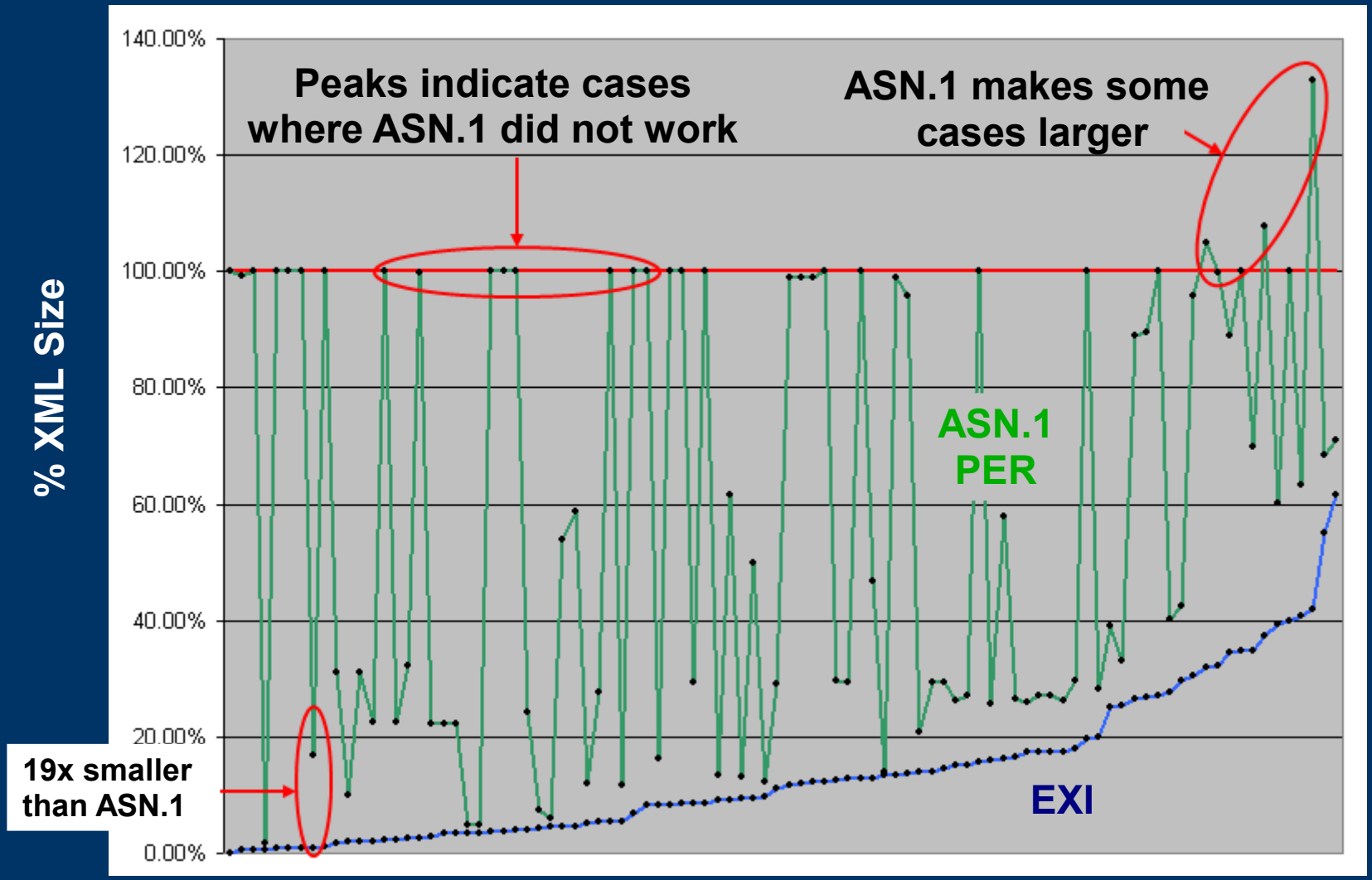
- EXI ドキュメント一覧
- メーリングリスト
- スケジュール

性能グラフ（ご参考）

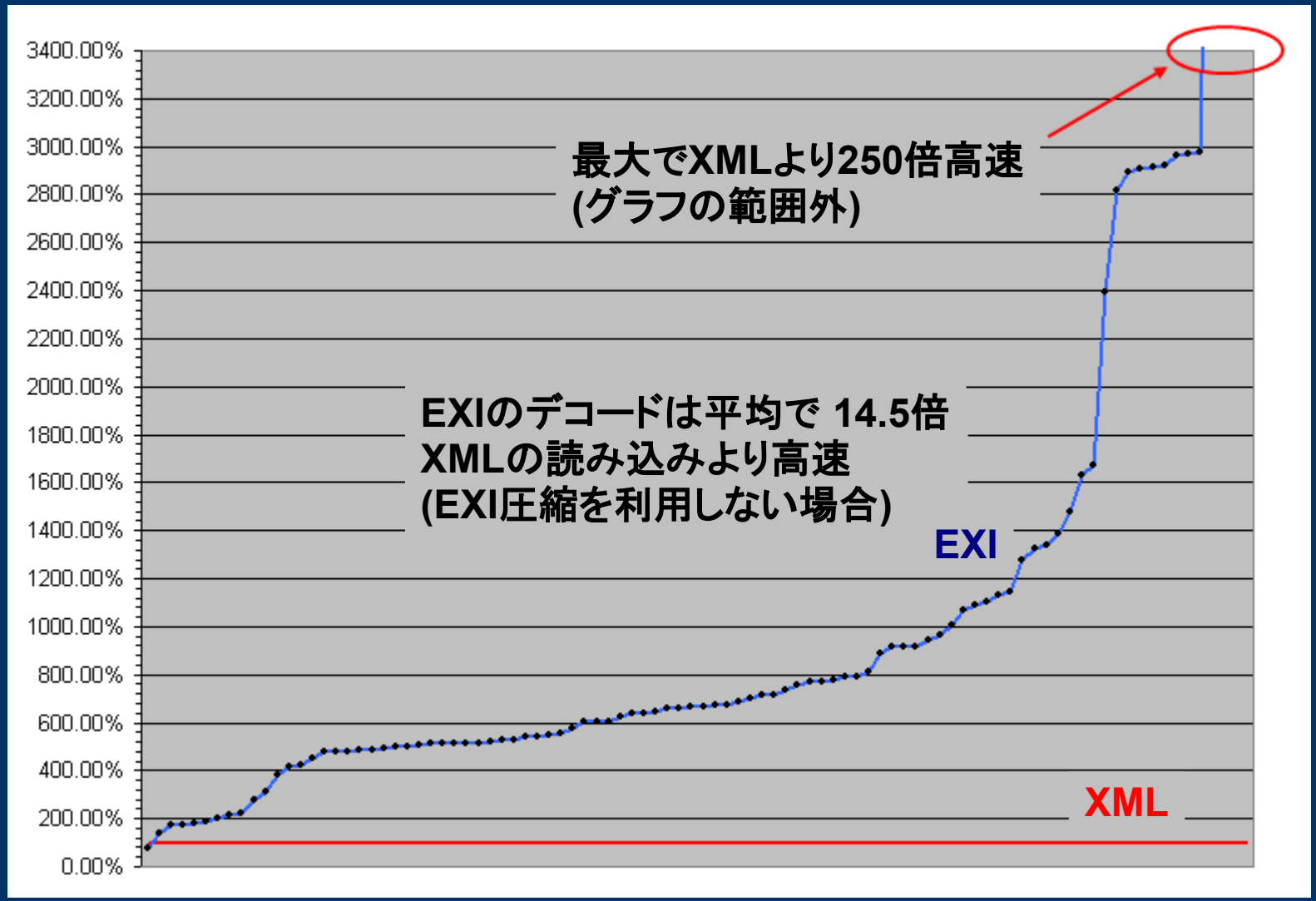
XML / GZIP / EXI 比較
ファイルサイズ



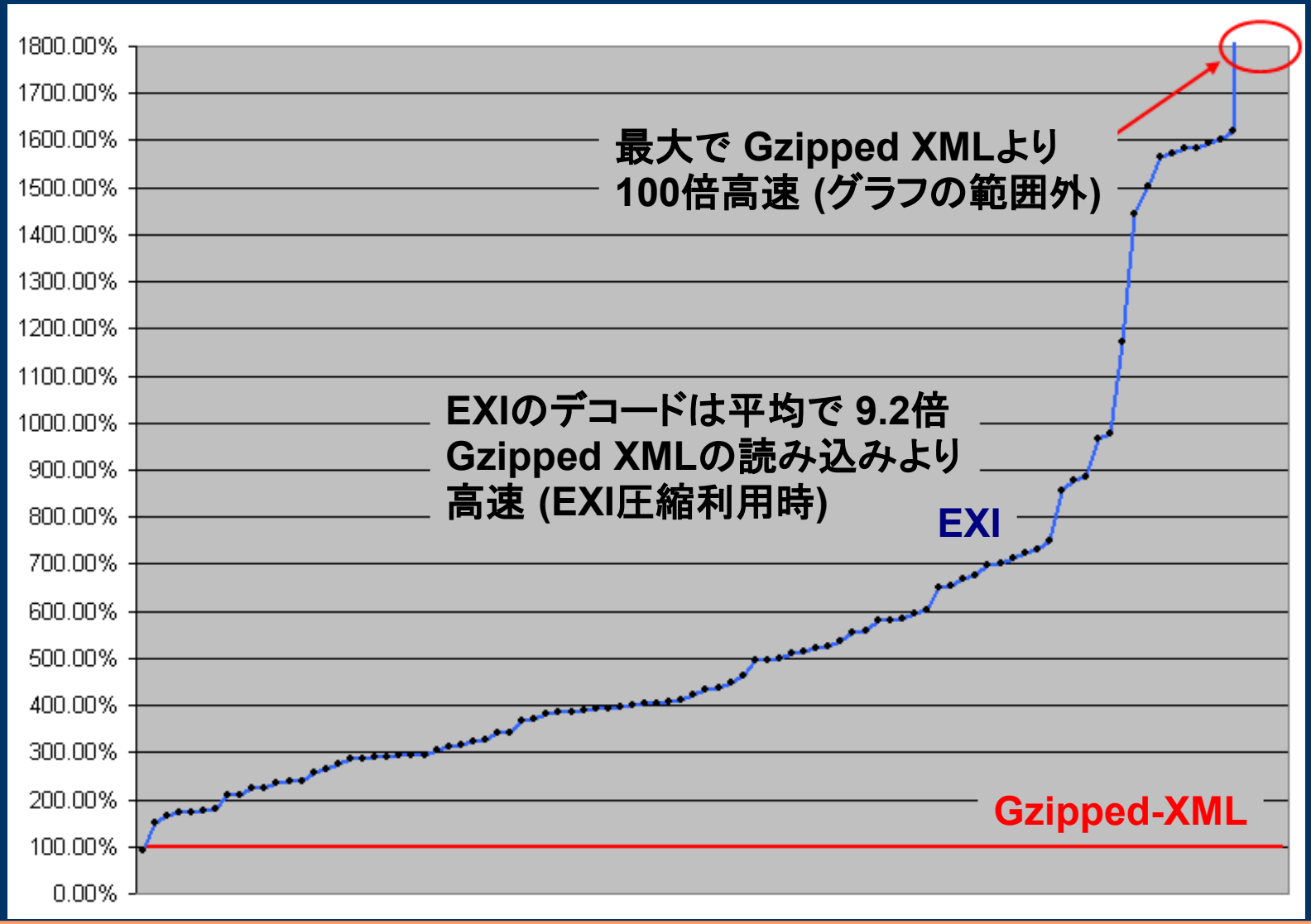
ASN.1 PER / EXI 比較
ファイルサイズ



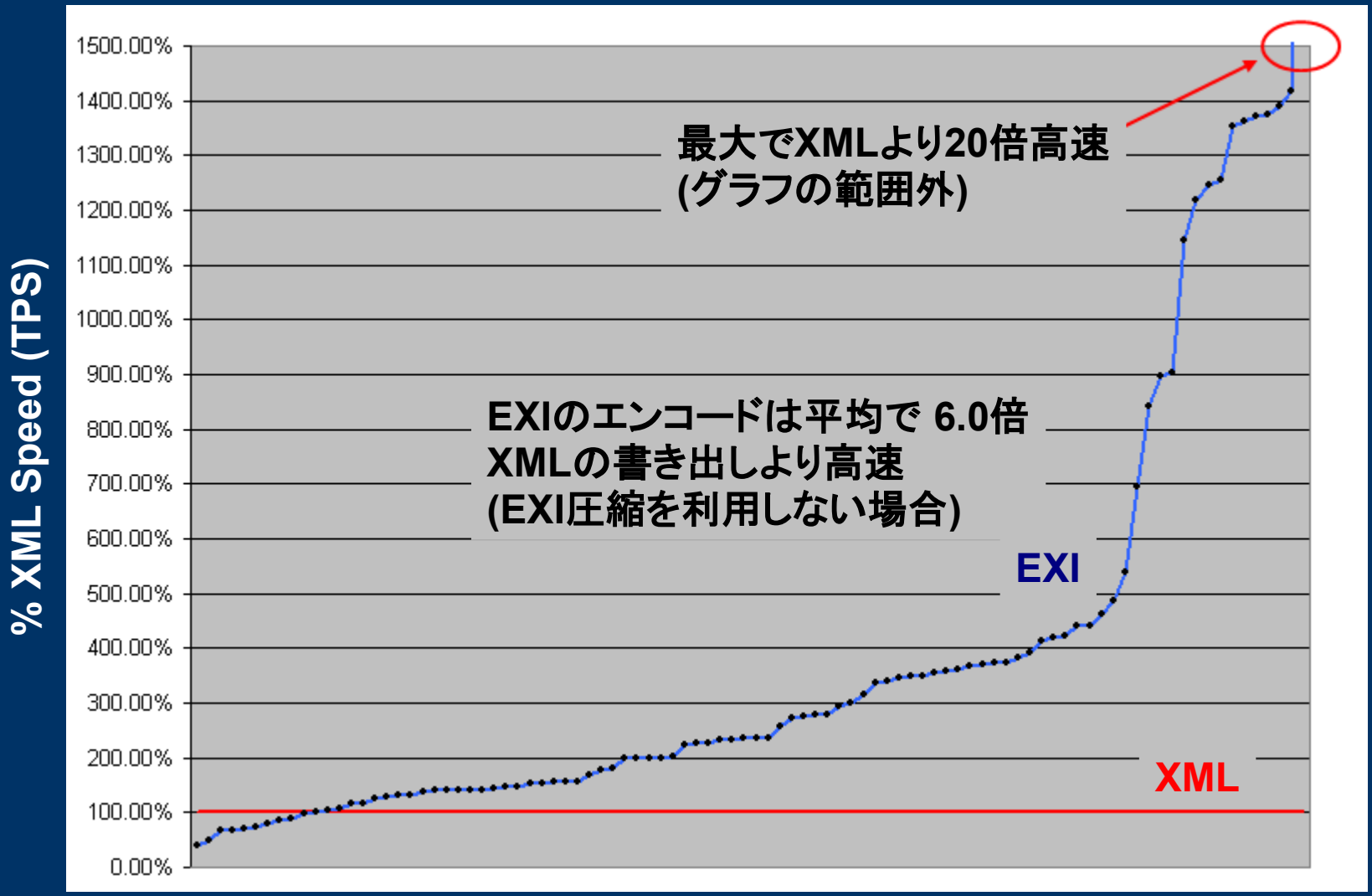
デコード処理性能比較（圧縮なし）



デコード処理性能比較（圧縮あり）



エンコード処理性能比較（圧縮なし）



エンコード処理性能比較（圧縮あり）