EPOCRATES®

XML encoding techniques for storing XML data on memory limited (mobile) devices.

David A. Lee
Senior member of the technical staff

XML encoding techniques for storing XML data on memory limited (mobile) devices.

- Introduction
  - Who is Epocrates ?
  - Common terminology
  - Application description
- Background
  - Characteristics of the application
  - Characteristics of the data
  - Why XML on the device ? A brief history
- Architecture
- Implementation
- Test Results

Introduction

- Who is Epocrates ?
- Common terminology
- Application description

Introduction
Who is Epocrates ?

- Epocrates is an industry leader in providing clinical references on handheld devices.
- 475,000 active subscribers
- Subscription based clinical publishing

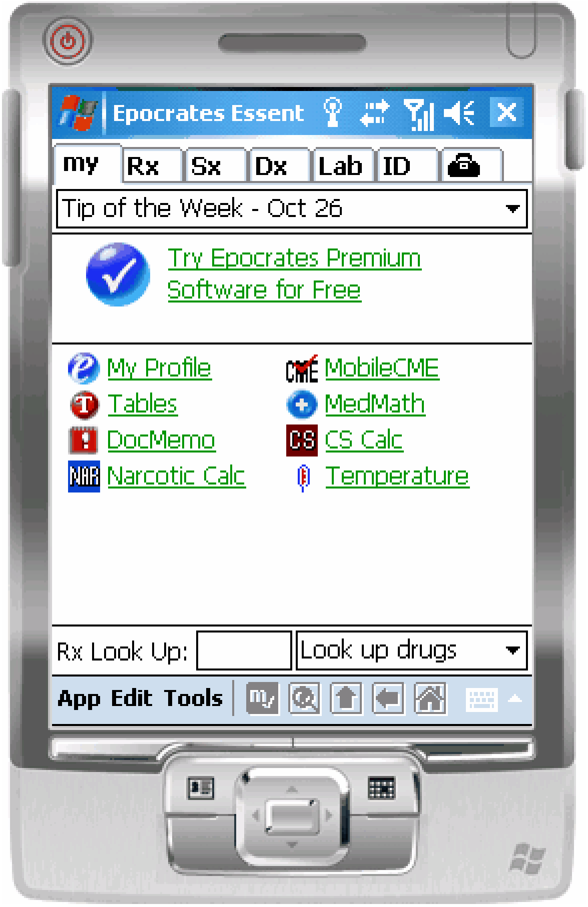
Introduction
Common Terminology

- **PDA** - "Personal Digital Assistant".
- **Monograph** - information describing a single drug, disease, lab test, preparation or other clinical entity.
- **Palm** – A PDA device running the Palm/OS operating system
- **PPC** - A PDA device running Microsoft's Pocket PC operating system.
- **Syncing** - The process of synchronizing a server's database with a PDA
- **PDB** - "Palm Database". A very simple variable length record format with a single 16 bit key index.

Introduction
Application description

"Essentials"

Handheld
Clinical
Reference

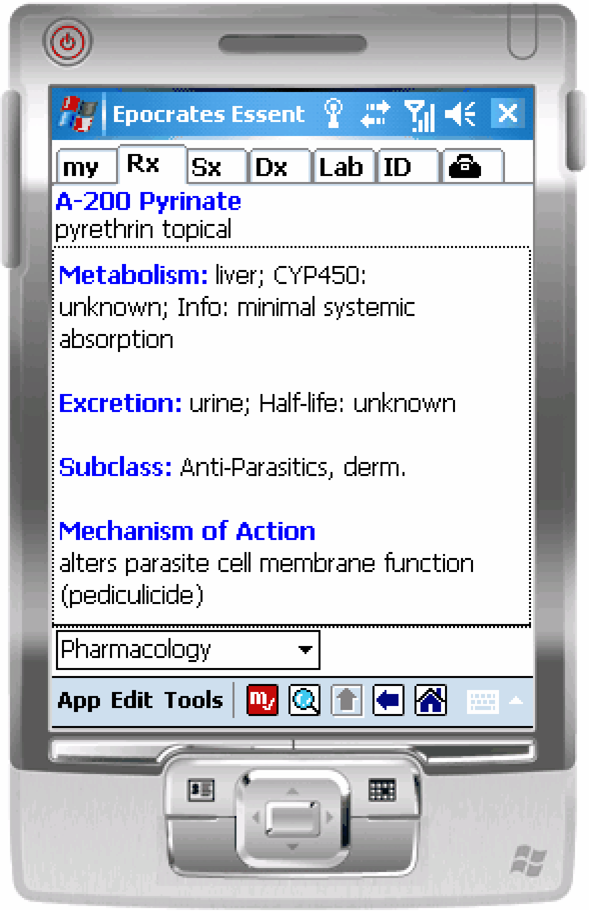


Tip of the Week - Oct 26

Try Epocrates Premium Software for Free

- My Profile
- Tables
- DocMemo
- Narcotic Calc
- MobileCME
- MedMath
- CS Calc
- Temperature

Rx Look Up: Look up drugs



A-200 Pyrinate
pyrethrin topical

Metabolism: liver; CYP450: unknown; Info: minimal systemic absorption

Excretion: urine; Half-life: unknown

Subclass: Anti-Parasitics, derm.

Mechanism of Action
alters parasite cell membrane function (pediculicide)

Pharmacology

EPOCRATES

Background “The Problem Space”

- Characteristics of the application
- Characteristics of XML data
- Why XML on device ? A brief history

Background
Characteristics of the Application

- Runs on handheld devices
- Limited memory capability (16MB typical)
- Slow CPU’s (33 – 400 Mhz)
  - Palm devices can be **effectively 2 Mhz due to “POSE”**
- UI response time critical
- Simple database
- Synchronization speed critical

Background
Characteristics of XML data

Different applications have different characteristics

- Disease reference XML Document containing many “monographs”. Approx 10 MB total
  - 1000 “monographs” 4 – 30 k each.
  - Both structural and “markup” type elements
- Message text
  - 100 messages avg. 1k each
  - Primarily “markup” elements
- Clinical References articles
  - 20 “pages” per article avg. 4k each
  - Primarily “markup” elements

Background
Why XML on the device ? A brief history

Initial implementation HTML and XML considered but:

- XML thought to be ...
  - Too Slow to parse
  - Too Large for devices
  - Too complicated
  - No XML ‘advocates’
- HTML unsuitable
  - Too hard to parse
  - Not appropriate set of markup features

An “RTF Like” markup was chosen

- /Bbold/btext/L/Anc32/aJump to app/l/Hheading/h

Background
Why XML on the device ? A brief history

Proprietary “RTF Like” markup hit a dead end ...

- Grew too complicated to parse
- No one could understand the code
- No one could understand the markup
- Extremely difficult to extend

XML was reconsidered !!!

- Extensible
- Well defined
- Maybe there was a way to optimize for mobile devices?

Architecture

- Focus on Parsing, not creating
- Simplify Schemas
- Split traditional parser into pieces
  - Server Piece - heavy front end
  - Client piece – light back end
  - Fixed “dictionary” if possible
- Efficient “SAX” Encoding
- Optional compression
- Pack for transport

Architecture
Focus on Parsing, not creating

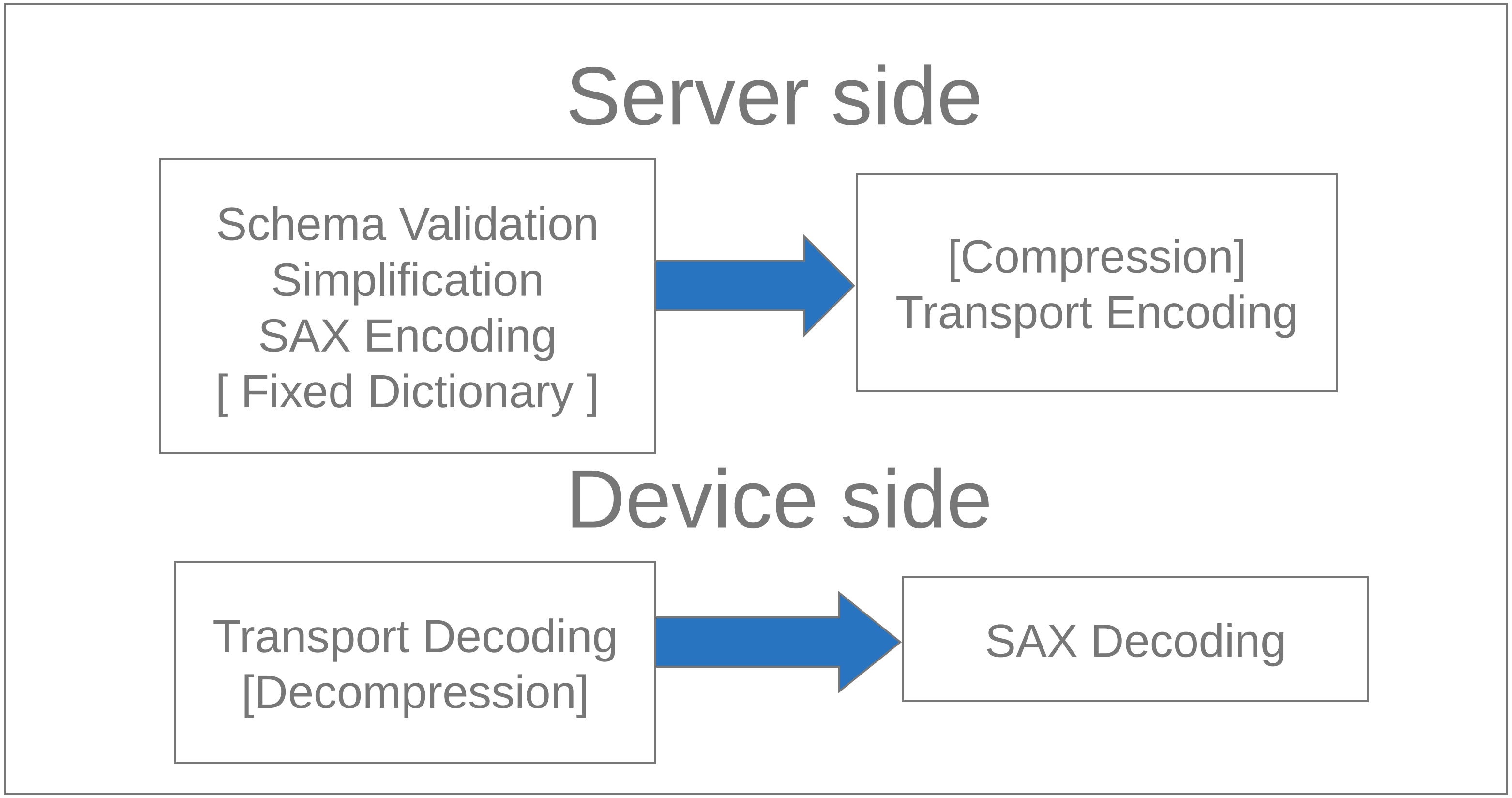
- Only Use Case is when XML is created on server
- Device parses XML only
- Does not need to create XML
  - Adding creation may be simple if schema is simple

Architecture
Simplify schemas

- Use simple schema for device side
  - Transform complex schemas to simple ones on the server
  - Minimize the number and complexity of elements, attributes
- Remove unnecessary features
  - Namespaces
  - Unicode in Element text (if possible)
  - Unicode in attribute values (if possible)

Architecture
Split parser into pieces

Full XML Parsing



Server side

Schema Validation
Simplification
SAX Encoding
[Fixed Dictionary]

[Compression]
Transport Encoding

Device side

Transport Decoding
[Decompression]

SAX Decoding

Architecture
Efficient “SAX Encoding”

- Use a SAX Parser to parse the XML into a stream of simple events and data
- Encode each event and data using a simple encoding scheme and a “Dictionary”
- Produces a stream which is very efficient to decode.

Architecture
Optional compression

- If Space is more important then speed, optionally compress.
- Open source “GZIP” compression works reasonably well on the encoded SAX stream.
- Decompression is costly (CPU) on the device so should be used sparingly
- Most compression algorithms work well only when given data > a few kb. (not good on small messages).

Architecture
Pack for transport

- Device databases usually have hard limits of record sizes (palm = 64k)
- Start, End markers and checksums may be useful
- Packing multiple streams into a larger record (before or after compression) for more efficient use of small documents

Implementation

- Server side (java)
  - Fixed Dictionary
  - SAX parser
  - SAX Encoding

- Device Side (C++)
  - Fixed Dictionary
  - SAX Decoder
  - C++ SAX Callbacks

Implementation
Fixed Dictionary

- Dictionary defines the mapping element and attributes (strings) to codes (integers).
- Dictionary typically would correspond to a single schema
- “Fixed” dictionary means the mapping is compiled in (implicit) and saves space, especially for small documents.

Implementation
Server – SAX Parser

- Java SAX parser
- Schema validation
- Schema Simplification
- Simplification and removal of unnecessary data.
  - Processing Instructions
  - Comments
  - Encoded Entities
  - Namespace elimination
- SAX Callbacks used to create a byte stream of encoded events + data (SAX Encoding)

Implementation
SAX Encoding Simplified ... (server)

```
static  int kEXT_START_DOC     = 0xFA ;
static  int kEXT_END_DOC       = 0xFB ;
static  int kSTART_ELEM   = 0xFC ;
static  int kEND_ELEM     = 0xFD ;
static  int kCHARACTERS   = 0xFE ;
```

```
startDocument()
        [kSTART_DOC]
startElement("name", null,null)
        [ELEM_ID]
startElement( "name" , attrs , nattr )
        [kSTART_ELEM][ELEM_ID][NATTR]
        [ATTR_ID]”value”\0[ATTR_ID|0x80][ENUM_ID] ...
characters( data , count )
        [kCHARACTERS]”string”\0
```

EPOCRATES®

Implementation
SAX Decoding Simplified … (client)

```
while( p < end  ){
  int c = *p++;
  switch( c ) {
   case kSTART_DOC :
       startDocument(); break;
   case kCHARACTERS :
       characters( p ) ; break ;
   case kSTART_ELEM :
        // start element
        ...
    default :
        startElement( c ) ;
}
```

Test Results

- Test Cases
- Test Devices
- Sample XML 12kbytes – largely text with markup
- Encoded Data Size
- Parsing Performance (wall time, on device)
- Normalized Parsing Performance (relative time, on device)

Test Results
Test Cases

XML	Text XML parsed with EXPAT
XML Compressed	Text XML compressed with GZIP Uncompressed then parsed with EXPAT
XText	XML SAX Encoded fixed dictionary Parsed with C++ SAX Decoder
XText Compressed	XML SAX Encoded and compressed with GZIP Uncompressed then parsed with C++ SAX Decoder

Test Results
Test Devices

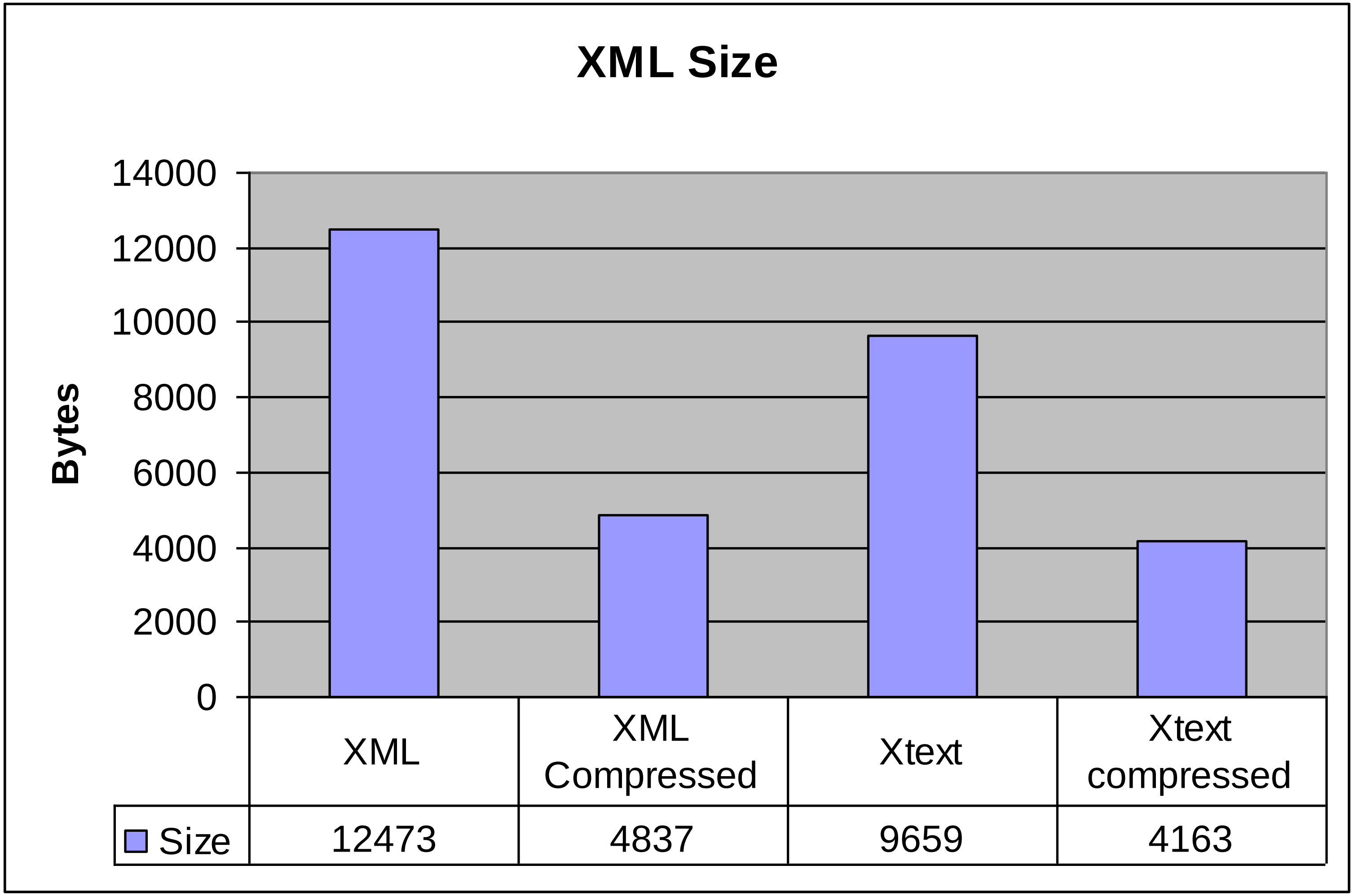
TE	Palm - Tungsten E 126 MHz
M500	Palm - M500 33 MHz
PPC	Pocket PC - HP IPAQ 4150 400 mhz

Test Results
Sample Doc (partial) – 12kbytes

```
<book>
<long_topic>
<id>TP0002</id>
<name>Abruptio placentae</name>
<content>
<basics>
<description> Premature separation of otherwise normally implanted placenta. Sher's grades:
1: Minimal or no bleeding; detected as retroplacental clot after delivery of viable fetus
2: Viable fetus with bleeding and tender irritable uterus
3: Type A with dead fetus and no coagulopathy; type B with dead fetus and coagulopathy
    (about 30% of grade 3's)
<systems_affected>
<system>Cardiovascular</system>
<system>Reproductive</system>
</systems_affected>
<genetics> N/A</genetics>
```

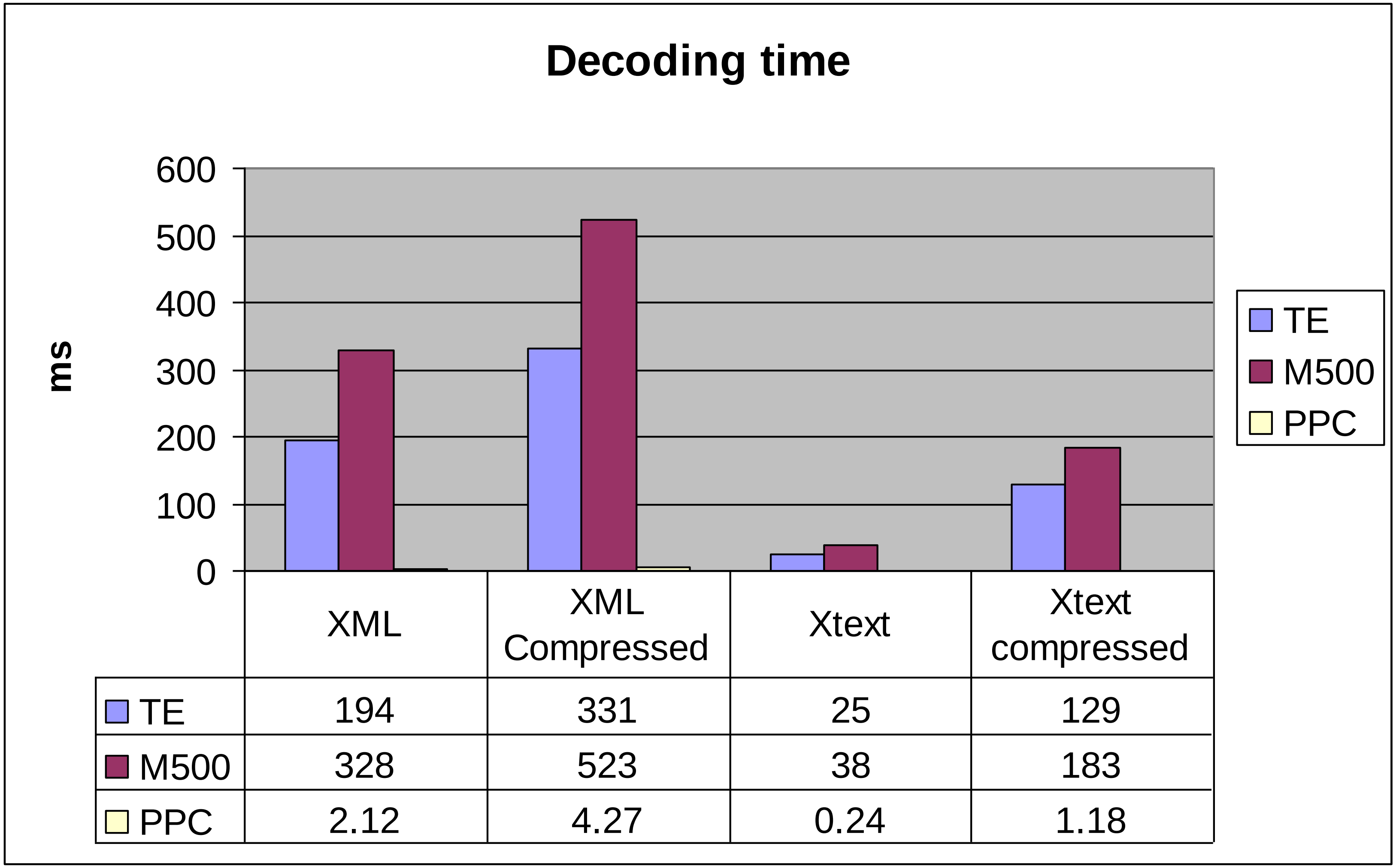
....   12 k bytes

Test Results
Size comparison of XML Encoding



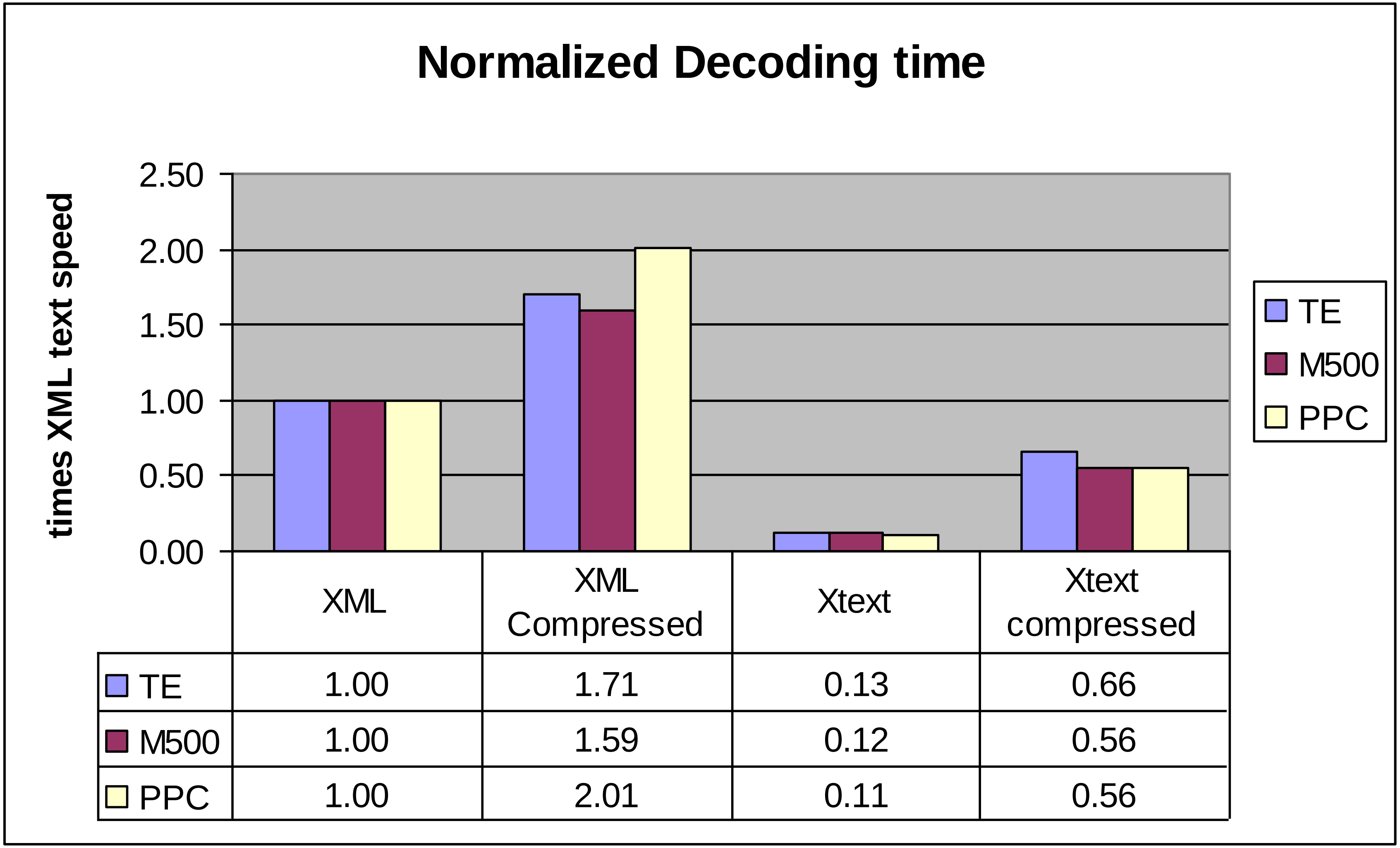
	XML	XML Compressed	Xtext	Xtext compressed
▨ Size	12473	4837	9659	4163

Test Results
Parsing Performance



	XML	XML Compressed	Xtext	Xtext compressed
🟦 TE	194	331	25	129
🟥 M500	328	523	38	183
□ PPC	2.12	4.27	0.24	1.18

Test Results
Normalized Parsing Performance


Normalized Decoding time

	XML	XML Compressed	Xtext	Xtext compressed
🟦 TE	1.00	1.71	0.13	0.66
🟥 M500	1.00	1.59	0.12	0.56
🟨 PPC	1.00	2.01	0.11	0.56

Summary

- Mobile devices have unique challenges
  - They CAN be solved !
- Split XML processing into server and client components
- On Server
  - Simplify XML documents
  - Encode efficiently
  - Optionally Compress
- On Device
  - Optionally Decompress
  - Efficient decoding
  - Avoid duplicate processing (what was already done on server)

Questions ?

Contact Info

David A. Lee

Epocrates, Inc

dlee@epocrates