BALTIMORE

Email: the Cinderella protocol?

Graham Klyne
Head of Strategic Research
Content Security Group

(This presentation has additional detail in the notes page view)

www.baltimore.com the global leader in e|security

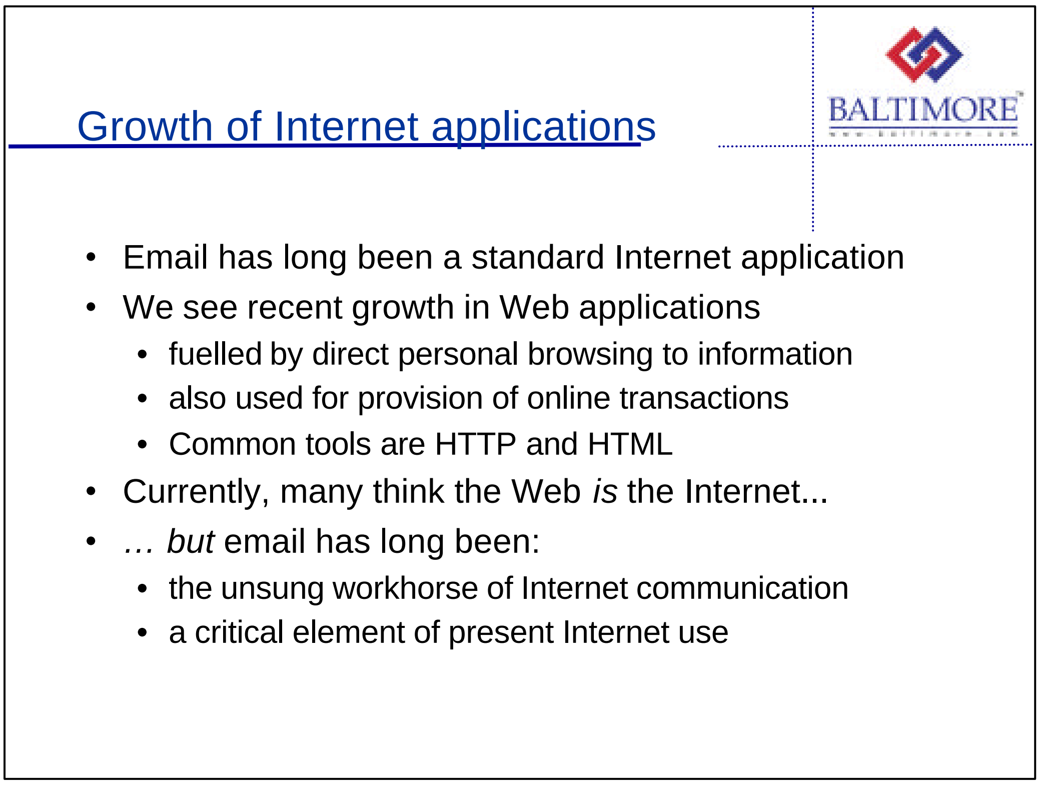
This presentation argues that email protocol principles could usefully be applied to a wider range of applications than just person-to-person messaging)

Why Cinderella?

- The “Cinderella” story is a well-known version of a very common folk tale. Cinderella is the less-favoured step-sister of her family, taken for granted and regarded as unimportant, but who eventually goes to the grand ball and catches the attention of her prince
- The email approach to data transfer, which has served so well for person-to-person communication over the past 20 years, may hold valuable lessons for modern Internet applications

From <http://members.aol.com/surlalune/frytales/cinderel/history.htm>:

Cinderella is easily one of the most well-known stories around the world. The themes from the story appear in the folklore of many cultures. Sources disagree about how many versions of the tale exist, with numbers ranging from 340 to over 1,500 if all of the picture book and musical interpretations are included. The tale has its own Aarne Thompson classification which is 510A. The tale always centers around a kind, but persecuted heroine who suffers at the hands of her step-family after the death of her mother. Her father is either absent or neglectful depending on the version. The heroine has a magical guardian who helps her triumph over her persecuters and receive her fondest wish by the end of the tale. The guardian is sometimes a representative of the heroine's dead mother. Most of the tales include an epiphany sparked by an article of clothing (usually a shoe) that causes the heroine to be recognized for her true worth.



Growth of Internet applications

- Email has long been a standard Internet application
- We see recent growth in Web applications
  - fuelled by direct personal browsing to information
  - also used for provision of online transactions
  - Common tools are HTTP and HTML
- Currently, many think the Web *is* the Internet...
- … *but* email has long been:
  - the unsung workhorse of Internet communication
  - a critical element of present Internet use

The past 5 years or so have seen an explosive growth in Web technologies, fuelled by direct access to a vast array of information using the common tools of HTTP and HTML. More recently (2-3 years), use of these technologies has been broadened to widespread provision of online transactions.

Currently, many people think the Web *is* the Internet.

Throughout this time, email has been the largely unsung workhorse of much Internet communication, especially person-to-person. Without email, much use of the Internet simply wouldn't happen:

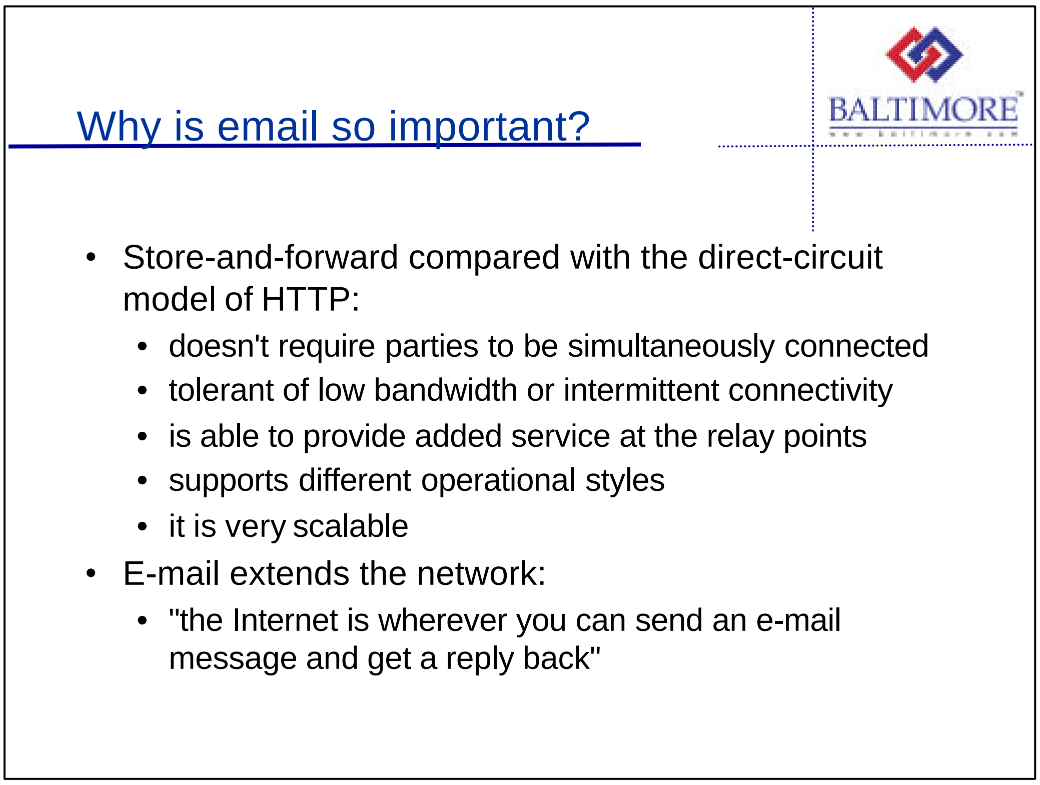
•Most person-to-person business communication

•Discussion lists (e.g. Internet technical development depends on email)

•Transaction confirmations

•Confirmation of location for security purposes

•etc.



Why is email so important?

- Store-and-forward compared with the direct-circuit model of HTTP:
  - doesn't require parties to be simultaneously connected
  - tolerant of low bandwidth or intermittent connectivity
  - is able to provide added service at the relay points
  - supports different operational styles
  - it is very scalable
- E-mail extends the network:
  - "the Internet is wherever you can send an e-mail message and get a reply back"

Unlike many other Internet application protocols, email does not require that communicating parties are simultaneously connected to the networking. There may be no end-to-end connectivity at any point in time.

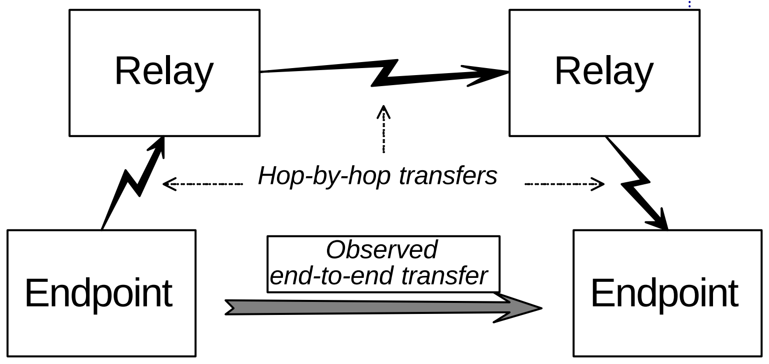
Email relies on a degree of capability within the network, in the form of “relays”, or MTAs, which can bear some of the responsibility for getting a message to its destination. The infrastructure, then, is better able to work around variations in bandwidth or transient breaks in connectivity, when the endpoint may be unavailable to participate in the process.

Additional services and capabilities can be provided at relay points: mailing lists, archiving, content checking, etc. Thus, different operational styles can be supported (sender initiated, polled collection, asynchronous notification, one-to-many distribution of information).

Email does not depend on a single point of coordination, and has been shown to be highly scalable.

Email is (relatively) easy to gateway. It allows effective communication to take place beyond the normal reach of the Internet. E.g. email to SMS messaging is a common service.

The email transfer model

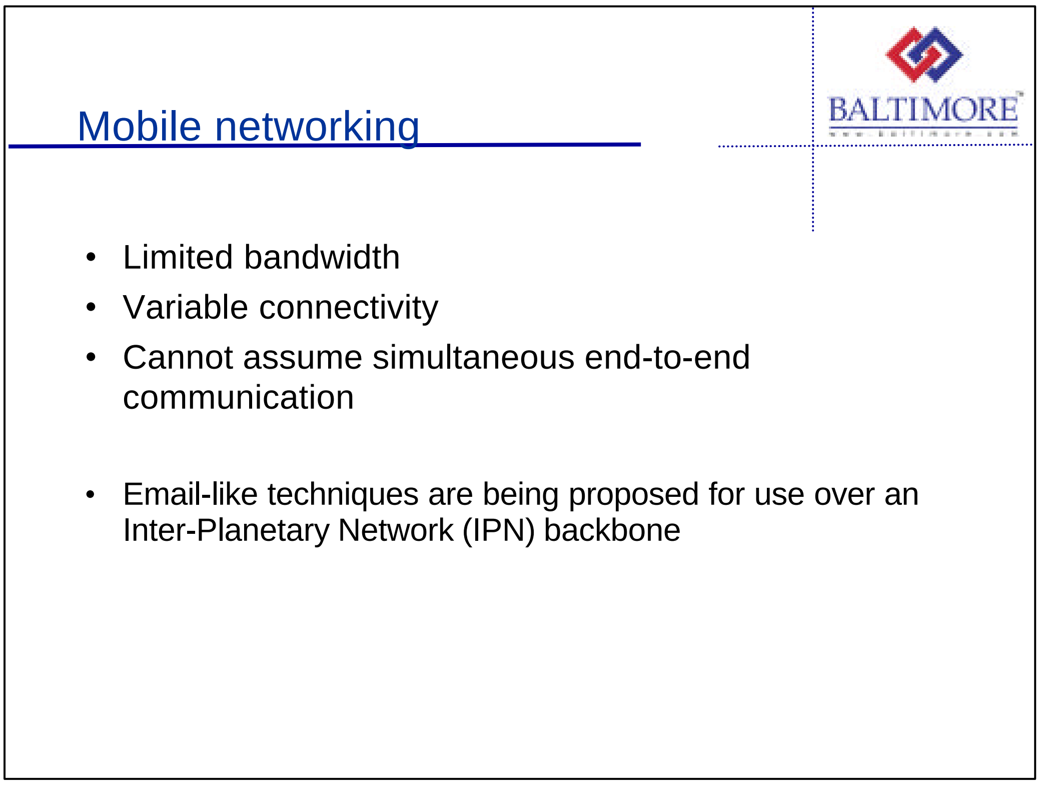


- Asynchronous message exchange
  - between: endpoints (User agents, MUA)
  - via: relays (Transfer agents, MTA)

The essence of the email transfer model is its use of relays that take on some degree of responsibility for moving a message to its destination.

Thus, the message can be moved through the network even when the sender, receiver or both are not actually able to participate in the process.

Contrast this with HTTP, and other direct-circuit models, which typically require both ends to be connected for a transaction to complete. (There are instances where HTTP is used to move data between intermediate stores, but I would argue that these are, to some extent, using TTP to implement aspects of the email transfer model.)



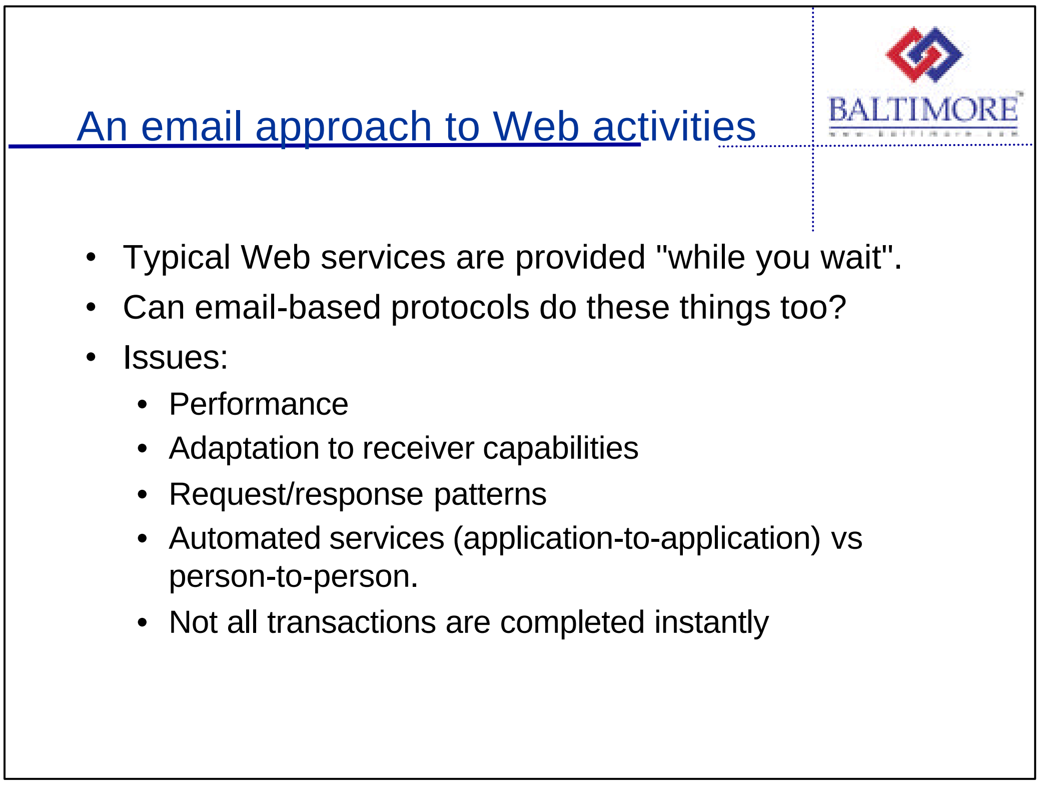
Mobile networking

- Limited bandwidth
- Variable connectivity
- Cannot assume simultaneous end-to-end communication

- Email-like techniques are being proposed for use over an Inter-Planetary Network (IPN) backbone

I submit that an asynchronous transaction capability will be particularly useful for mobile devices -- where connectivity may be highly variable.

Vinton Cerf and others have noted here at INET 2001 that applications over an Interplanetary Network (IPN) backbone cannot depend on end-to-end connections. The use of email-like techniques is being proposed to overcome these constraints.



An email approach to Web activities

BALTIMORE

- Typical Web services are provided "while you wait".
- Can email-based protocols do these things too?
- Issues:
  - Performance
  - Adaptation to receiver capabilities
  - Request/response patterns
  - Automated services (application-to-application) vs person-to-person.
  - Not all transactions are completed instantly

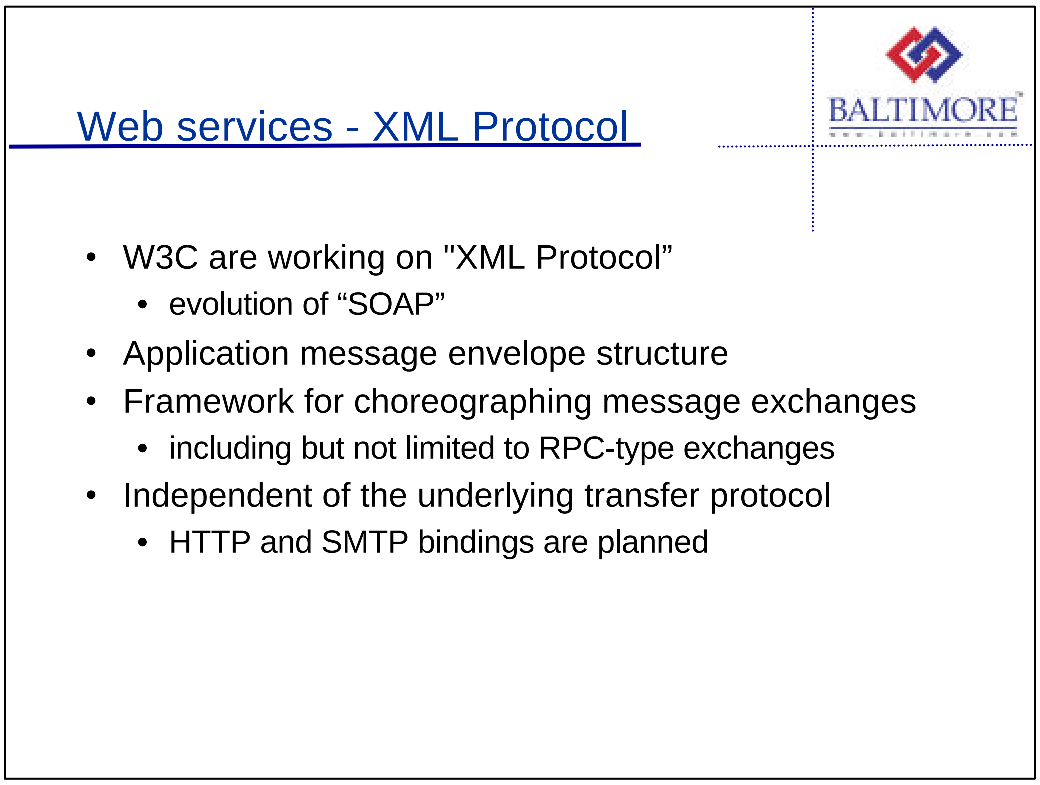
Contrary to popular belief, email is not inherently slow; its performance has more to do with service provisioning than protocol limitations... but traditional email protocols provide no way for a sender to determine time limits for delivery. Recent work changes this (c.f. DELIVERBY and ongoing Internet fax work).

Current email requires the sender to know in advance whether or not the receiver can process some particular data. This is generally OK for person-to-person communication where the people concerned can communicate capabilities "out of band", but is not good for more automated forms of message transfer. Current Internet fax work is addressing this issue.

Request-response patterns can be used with email (c.f. MIME message/external-body, and ongoing Internet fax work)

Email has focused on person-to-person communication. Work on the APEX protocol aims to apply the "relay mesh" paradigm to application-to-application communications. The basic relay mesh is very simple, but provides hooks to support application determination of message-passing behaviours (reliability, timeliness, etc.).

Not all transactions need to be instant. Where existing e-commerce systems cannot complete a transaction instantly, they commonly fall back on email to complete the process; e.g. final confirmation.



Web services - XML Protocol

BALTIMORE

- W3C are working on "XML Protocol"
  - evolution of “SOAP”
- Application message envelope structure
- Framework for choreographing message exchanges
  - including but not limited to RPC-type exchanges
- Independent of the underlying transfer protocol
  - HTTP and SMTP bindings are planned

W3C are working on a specification for an "XML Protocol", also known as SOAP. This is not so much a protocol as an application message envelope structure and framework for choreographing message exchanges (including but not limited to RPC-type exchanges).

It is being designed to be independent of the underlying transfer protocol; both HTTP and SMTP bindings are currently planned.

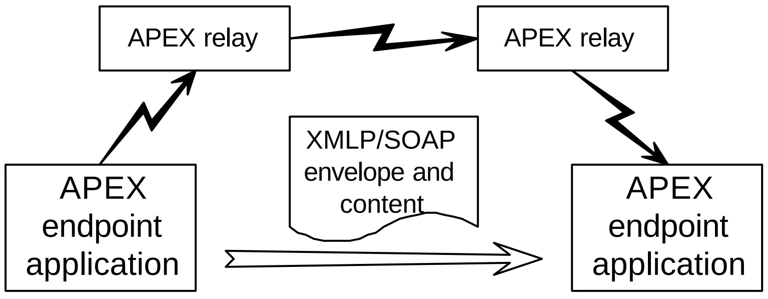
I contend that APEX would be an ideal message transfer substrate for asynchronous XML protocol interactions, providing as it does the advantages of a very simple relay-based substrate designed with a view to applications other than person-to-person messaging.

A common application envelope structure (one could think of doing for application-to-application messaging what RFC822 has done for person-to-person email) would mean that there are real possibilities for integrating asynchronous and synchronous message handling into a common application framework.

Of course, within this framework, person-to-person communication (e.g. email and instant messaging) can be offered as particular applications.

Application messaging - APEX

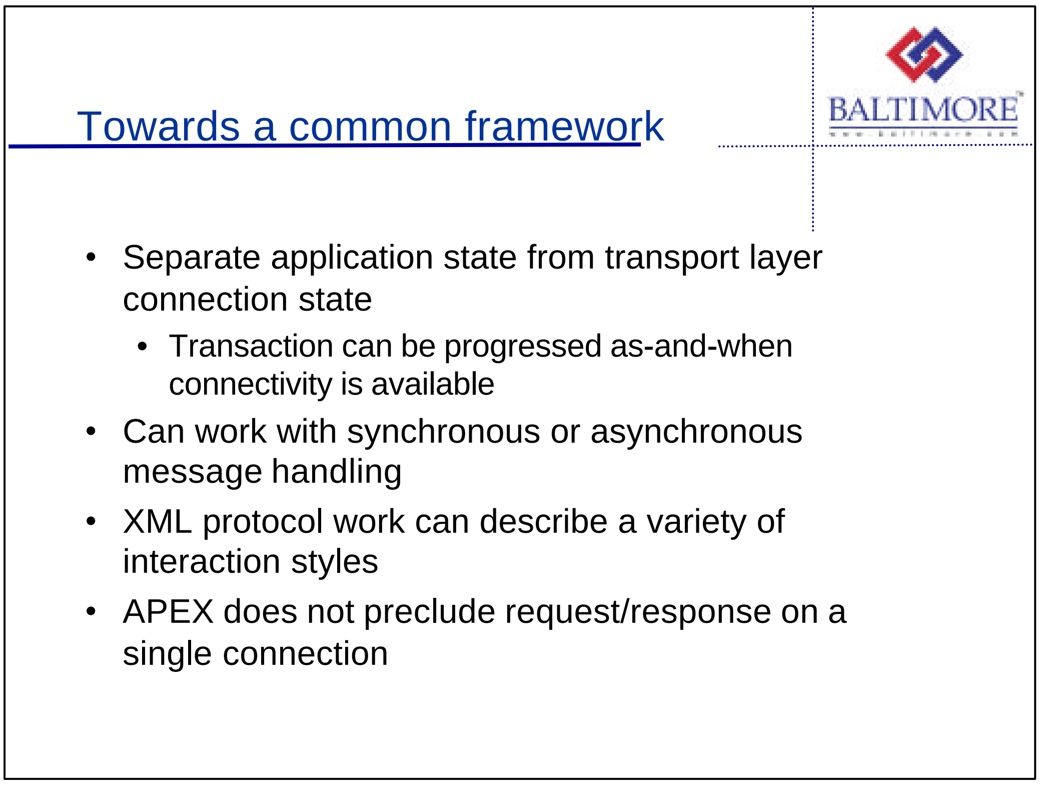
- Ongoing IETF design activity
  - based on email relay-mesh architecture
  - more application oriented
- APEX as a possible substrate for XML Protocol:


APEX relay
APEX relay
XMLP/SOAP envelope and content
APEX endpoint application
APEX endpoint application

The APEX protocol is being developed in the IETF. It is very much based on the basic architecture of Internet email (SMTP, etc.) but has been designed as a general substrate for application data rather than just person-to-person messaging.

The default message transfer semantics are minimal (instant, best effort), but extension facilities are provided to allow additional semantics to be applied (e.g. confirmation of response, store-and-forward, timeliness).

I view APEX as an ideal substrate for XML protocols: sufficiently lightweight for efficient application-to-application communication, but flexible enough to deal with challenging network conditions.



Towards a common framework

BALTIMORE

- Separate application state from transport layer connection state
  - Transaction can be progressed as-and-when connectivity is available
- Can work with synchronous or asynchronous message handling
- XML protocol work can describe a variety of interaction styles
- APEX does not preclude request/response on a single connection

Protocols like HTTP force elements of the application state to be bound to the underlying transport connection state. HTTP, for example, requires that a response is received on the same connection that is used to send the corresponding request.

It is not uncommon for an HTTP transaction to be completely abandoned if connectivity cannot be maintained for its entire duration.

By separating transaction state from underlying protocol state, the applications can arrange to progress a transaction as-and-when connectivity is available.

In many cases, a connection identifier (socket, handle or other token) is part of some larger state that applications have to maintain in any case. Taking the connection out of the application state means that multiple connections can be used to conduct a transaction… if connectivity is lost then the transaction can be resumed when it comes available again.

XML protocol is being designed to permit a number of different interaction styles: one-way messaging and request/response are two common cases. Others might involve more complex message interactions. Using APEX as a substrate for XML protocol could, in principle support any chosen pattern of interaction.

Conclusion

- The successes of email can be applied to applications other than person-to-person messaging
- There are many features of the email protocol model that are particularly well-suited to commercial transactions using mobile devices (m-commerce)
- Maybe email, or application messaging, can come to the e-commerce ball after all?

BALTIMORE

www.baltimore.com