Network Working Group
Internet Draft
<draft-ietf-calsch-imp-guide-02.txt>
November 24 2000
Expires: May 24 2001

Bob Mahoney/MIT
Alexander Taler
George Babics/Steltor

Implementors' Guide to Internet Calendaring

Status of this Memo

Abstract

This document describes the various internet calendaring and scheduling standards and drafts and the relationships between them. It's intention is to provide a context for these documents, assist in their understanding, and potentially help implementors in the design of their internet calendaring and scheduling systems. The standards addressed are RFC 2445 (iCalendar), RFC 2446 (iTIP), and RFC 2447 (iMIP). The draft addressed is "Calendar Access Protocol" (CAP).

[Note: in the past there has been some discussion as to whether iRIP was a live effort, given that interest has waned and some functionality has been moved to CAP. Its status will be discussed further.]

This document also describes issues and problems that are not solved by these protocols, and could be targets for future work.

Status of this Memo

1. Introduction

The calendaring and scheduling protocols are intended to provide for the needs of individuals attempting to obtain calendaring information and schedule meetings across the internet, organizations attempting to provide calendaring information on the internet, as well as organizations looking for a calendaring and scheduling solution to deploy internally.

It is the intent of this document is to provide a context for the calendar standard and draft documents, assist in their understanding, and potentially help implementors in the design of their internet calendaring and scheduling systems.

Problems not solved by these protocols, as well as security issues to be kept in mind, are discussed at the end of the document.

1.1 Terminology

This memo uses much of the same terminology as [ICAL], [ITIP], [IMIP], [IRIP] and [CAP]. The following definitions are provided as introductory, the definitions in the protocol specifications are the canonical ones.

Calendar
: A collection of events, todos, journal entries, etc. A calendar could be the content of a person or a resource's agenda; it could also be a collection of data serving a more specialized need. Calendars are the basic storage containers for calendaring information.

Calendar Access Rights
: A set of rules for a calendar describing who may perform which operations on that calendar, such as reading and writing information.

Calendar Service
: A running server application which provides access to a collection of calendars.

Calendar Store
: A data store of a calendar service. A calendar service may have several calendar stores, and each store may contain

several calendars, as well as properties and components outside of the calendars.

Calendar User
: An entity (often a human) that accesses calendar information.

Calendar User Agent (CUA)
: Software used by the calendar user that communicates with calendar services to provide the user access to calendar information.

Component
: A piece of calendar data such as an event, a todo or an alarm. Information about components is stored as properties of those components.

Delegate
: Is a calendar user (sometimes called the delegatee) who has been assigned participation in a scheduled calendar component (e.g., VEVENT) by one of the attendees in the scheduled calendar component (sometimes called the delegator). An example of a delegate is a team member told to go to a particular meeting.

Designate
: Is a calendar user who is authorized to act on behalf of another calendar user. An example of a designate is an assistant.

Local Store
: A CS which is on the same platform as the CUA.

Property
: A property of a component, such as a description or a start time.

Remote Store
: A CS which is not on the same platform as the CUA.

1.2 Concepts and Relationships

iCalendar is the Language to be used in calendar events.
iTIP is how you use the language.
iMIP is further definition for use over email.
iRIP is the language used over a real-time transport
CAP is how to use the language, in real-time, to access a calendar server

Another way to put it is as follows:
iCalendar are the words
iTIP is the grammar book or the "Rosetta Stone".
iMIP is "expressing it in email terminology" an EMAIL dictionary
CAP/iRIP is "expressing it for use in a Real Time transport"

A comparison with email:
RFC822 in email: iRIP in Calendaring (scheduling not booking)
POP/IMAP in email: CAP in calendaring
iMIP uses RFC822
RFC822 is a wrapper for email: iTIP is a wrapper for

calendaring objects

2. Requirements

2.1 Fundamental Needs

The following examples illustrate people's and organizations' basic calendaring and scheduling needs:

a] A doctor wishes to keep track of all his appointments.

Need: Read and manipulate one's own calendar with only one CUA.

b] A busy musician wants to maintain her schedule on an internet-based agenda which she can access from anywhere.

Need: Read and manipulate one's own calendar.

c] A software development team wishes to share agenda information by using a group scheduling product in order to more effectively schedule their time.

Need: Share calendar information with users using the same calendar service.

d] A teacher wants his students to be able to book time slot during his office hours.

Need: Schedule calendar events and todos with users using the same calendar service.

e] A movie theatre wants to publish its schedule so that prospective customers can easily access it.

Need: Share calendar information with users using other calendar services, possibly from different vendors.

f] A social club wants to be able to organize events more effectively by booking time with its members.

Need: Schedule calendar events and todos with users using other calendar services, possibly from different vendors.

2.2 Protocol requirements

The first four needs can be satisfied through proprietary solutions, but the last two cannot. From these needs we can establish that protocols are required for accessing information in a calendar store,

and for scheduling events and todos. In addition these protocols require a data format for representing calendar information.

These roles are filled by the following protocol requirements.

- [ICAL] is the data format

[ICAL] provides data format for representing calendar information which the other protocols can use. [ICAL] can also be used in other contexts such as a drag and drop format or an export/import format.

All the other protocols depend on [ICAL], so all elements of a standards-based calendaring and scheduling systems will have to interpret [ICAL].

For example the following describes an event:

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//ABC Corporation//NONSGML My Product//EN
BEGIN:VEVENT
ORGANIZER;CN=Jane Doe:MAILTO:jane@bar.com
DTSTART:20000905T120000Z
SUMMARY:Lunch
DTEND:20000905T130000Z
ATTENDEE;CN=John Smith:MAILTO:john@foo.com
ATTENDEE;CN=Jane Doe:MAILTO:doe@bar.com
END:VEVENT
END:VCALENDAR
```

- [ITIP] is the scheduling protocol

[ITIP] describes the messages used to schedule calendar events. These messages are represented in [ICAL], and have semantics that include such things as being an invitation to a meeting, an acceptance of an invitation or the assignation of a task.

[ITIP] messages are used in the scheduling work flow, where users exchange messages in order to organize things such as events and todos. CUAs generate and interpret [ITIP] messages at the direction of the calendar user.

With [ITIP] one can create, modify, delete, reply to,

counter, and decline counters to, the various [ICAL] components. Furthermore, one can also request the freebusy time of other people.

For example, to invite a user to the above event, one can send a message like this one:

```
BEGIN:VCALENDAR
METHOD:REQUEST
VERSION:2.0
PRODID:-//ABC Corporation//NONSGML My Product//EN
BEGIN:VEVENT
ORGANIZER;CN=Jane Doe:MAILTO:jane@bar.com
DTSTART:20000905T120000Z
SUMMARY:Lunch
DTEND:20000905T130000Z
ATTENDEE;CN=John Smith:MAILTO:john@foo.com
ATTENDEE;CN=Jane Doe:MAILTO:doe@bar.com
END:VEVENT
END:VCALENDAR
```

The user, John Smith, can send a reply using the REPLY method.

[ITIP] is transport-independent, but has two specified transport bindings, [IMIP] is a binding to email and

[IRIP] is a real-time binding. In addition [CAP] will provide a second real-time binding of [ITIP], allowing CUAs to perform calendar management as well as scheduling over a single connection.

For example, sending the above request using iMIP would look like:

```
From: jane@bar.com
To: john@foo.com
Subject: Lunch
Mime-Version: 1.0
Content-Type:text/calendar; method=REQUEST;charset=US-ASCII
Content-Transfer-Encoding: 7bit

BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//ABC Corporation//NONSGML My Product//EN
BEGIN:VEVENT
```

```
ORGANIZER;CN=Jane Doe:MAILTO:jane@bar.com
DTSTART:20000905T120000Z
SUMMARY:Lunch
DTEND:20000905T130000Z
ATTENDEE;CN=John Smith:MAILTO:john@foo.com
ATTENDEE;CN=Jane Doe:MAILTO:doe@bar.com
END:VEVENT
END:VCALENDAR
```

Both CUAs and calendar services may have [ITIP] interpreters.

- [CAP] is the calendar management protocol

[CAP] describes the messages used to manage calendars. These messages are represented in [ICAL], and have semantics such as being a search for data, being data in response to a search or the being the creation of a meeting.

[CAP] describes the messages used to manage calendars on a calendar store.

These messages are represented in [ICAL]. With these messages one can do the operations in [ITIP] and other operations relating to a calendar store. These operations include, search, creating calendars, specifying calendar properties, and being able to specify access rights to one's calendars.

[CAP] also provides a real-time binding for the calendar management messages. Although other bindings, such as an email binding, could be defined, this is not done because it is inappropriate for this protocol.

For example, one can schedule the above meeting using CAP:

```
C:SENDATA
C:CONTENT-TYPE:text/calendar; method=CREATE;charset=US-ASCII
C:content-transfer-encoding: 7bit
C:BEGIN:VCALENDAR
C:VERSION:2.0
C:PRODID:-//ABC Corporation//NONSGML My Product//EN
C:TARGET:cap://cal.example.com/johns
```



```
C:TARGET:janed
C:METHOD:CREATE
C:BEGIN:VEVENT
```

```
C:ORGANIZER;CN=Jane Doe:MAILTO:jane@bar.com
C:DTSTART:20000905T073000Z
C:SUMMARY:Lunch
C:DTEND:20000905T120000Z
C:END:VEVENT
C:END:VCALENDAR
C: .
S: 2.0
S: Content-Type:text/calendar; method=RESPONSE;
S:
S: BEGIN:VCALENDAR
S: VERSION:2.1
S: METHOD:RESPONSE
S: BEGIN:VEVENT
S: TARGET::cap://cal.example.com/johnd
S: REQUEST-STATUS:2.0
S: END:VEVENT
S: END:VCALENDAR
S: Content-Type:text/calendar; method=RESPONSE;
S:
S: BEGIN:VCALENDAR
S: VERSION:2.1
S: METHOD:RESPONSE
S: BEGIN:VEVENT
S: TARGET::cap://cal.example.com/janed
S: REQUEST-STATUS:2.0
S: END:VEVENT
S: END:VCALENDAR
S: .
```

Note that "C" indicates the data sent by the client, and "S" the data sent by the server. Furthermore, CAP is still a draft, thus the details of one can create such an event may change.

The dependencies between the different protocols are as follows:

iCalendar is the language set used to describe/specify calendaring events or operations.

When specified using correct iCalendar grammar, we refer to these event representations or operation requests as " calendar object representations

There are two main methodologies for communicating iCalendar objects:

1) Via a store-and-forward mechanism (usually email), using the iMIP specification.

2) Via an on-the-wire mechanism (a directly connected state,

however briefly), using the CAP specification.

A system may implement the first methodology only. The second one is dependent on iTIP. It requires understanding of iTIP and the ability to communicate with other CAP servers using iTIP. Since, currently, iMIP is the only binding of iTIP, the second method is also dependent on iMIP.

Additionally, the iTIP specification describes a transport-independent grammar for communicating between systems. The iMIP specification utilizes iTIP to express iCalendar objects.

3. Solutions

3.1 Examples

Returning to the examples of section 2.1, they can be solved using the protocols in the following ways:

a] The doctor can use a proprietary CUA with a local store, and perhaps use [ICAL] as a storage mechanism. This would allow the doctor to easily import his store into another application that supports [ICAL].

b] The musician who wishes to access her agenda from anywhere can use a [CAP] enabled calendar service accessible through the internet. She can then use whichever [CAP] clients are available to access the data.

A proprietary system could also be employed which provides access through a web-based interface, but the use of [CAP] would be superior in that it would allow the use of third party tools, such as PDA synchronization tools.

c] The development team can use a calendar service which supports [CAP] and then each member can use a [CAP]-enabled CUA of their choice.

Alternatively, each member could use an [IMIP]-enabled CUA, and they could book meetings over email. This solution has the drawback that it is difficult to examine the other agendas, making organizing meetings more difficult.

Proprietary solutions are also available, but they require that all people use clients by the same vendor, and disallow

the use of third party applications.

d] The teacher can set up a calendar service, and have students book time through any of the [ITIP] bindings. [CAP] or [IRIP] provide real-time access, but could require additional configuration. [IMIP] would be the easiest to configure, but may require more email processing.

If [CAP] access is provided then determining the state of the teacher's schedule is straightforward. If not, this can be determined through [ITIP] free-busy requests. Non-standard methods could also be employed, such as serving up ICAL, HTML, XML through HTTP.

A proprietary system could also be used, but would require that all students be able to use software from a specific vendor.

e] For publishing a movie theatre's schedule [CAP] provides the most advanced access and search capabilities. It also allows easy integration with its customer's calendar systems.

Non-standard methods such as serving data over HTTP could also be employed, but would be harder to integrate with

customer's systems.

Using a completely proprietary solutions would be very difficult since it would require every user to install and use proprietary software.

f] The social club could distribute meeting information in the form of [ITIP] messages. This could be done over email using [IMIP], or [IRIP] depending on the recipient. Meeting invitations, as well as a full published agenda could be distributed.

Alternatively, the social club could provide access to a [CAP] enabled calendar service, however this solution would be more expensive since it requires the maintenance of a server.

3.2 Systems

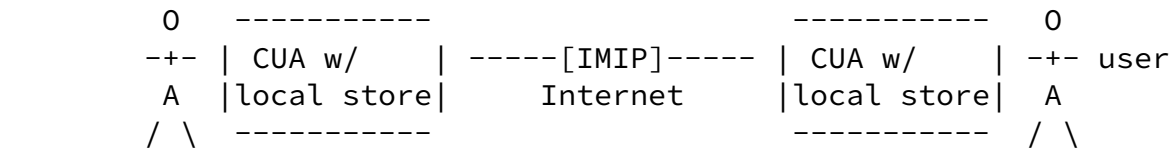
The following diagrams illustrate possible example systems and usage of the protocols.

3.2.1 Standalone single-user system

A single user system that does not communicate with other systems need not employ any of the protocols. However, it may use [ICAL] as a data format in some places.

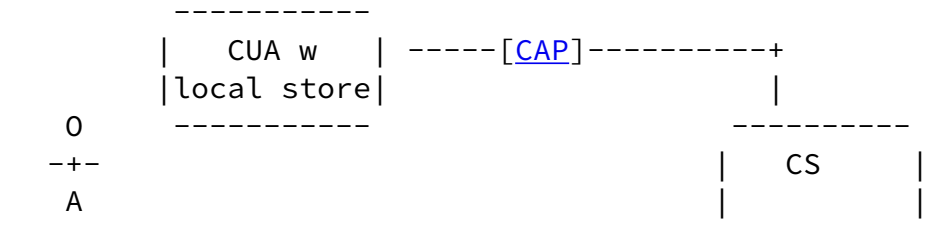
3.2.2 Single-user systems communicating

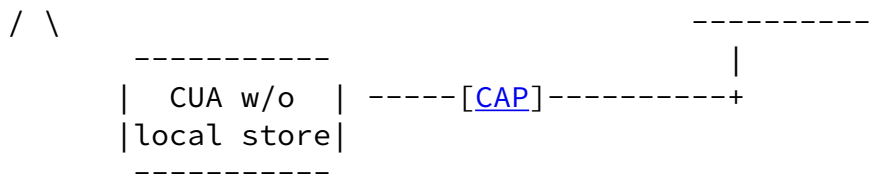
Users with single-user systems may schedule meetings with each other using [ITIP]. The easiest binding of [ITIP] to use is [IMIP], since it messages can be held in their mail queue, which we assume to already exist. [IRIP] or [CAP] would require at least one user to run a listening server.



3.2.3 Single-user with multiple CUA

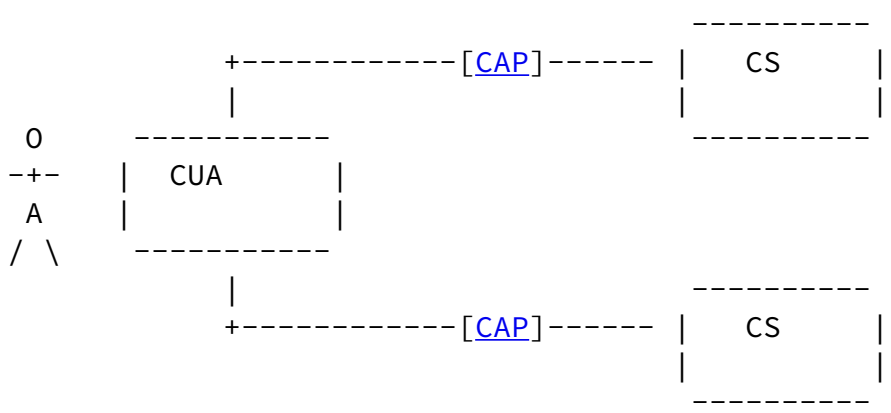
A single user may use more than one CUA to access his or her calendar. The user may use a PDA, a web client, a PC, or some other device, depending an accessibility. Some of these clients may have local stores and others may not. If they do, then they need to ensure that the data on the CUA is synchronized with the data on the CS.





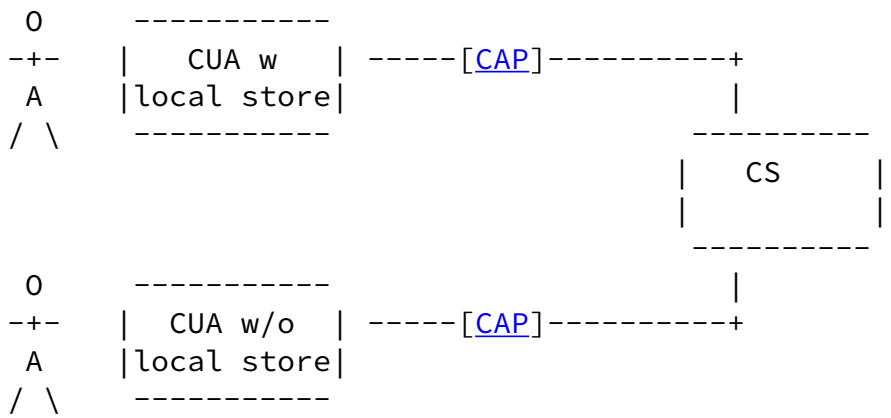
3.2.4 Single-user with multiple calendars

A single user may have many independent calendars. One may be work related, another for personal use. The CUA may or may not have a local store. If it does, then it needs to ensure that the data on the CUA is synchronized with the data on both of the CS.



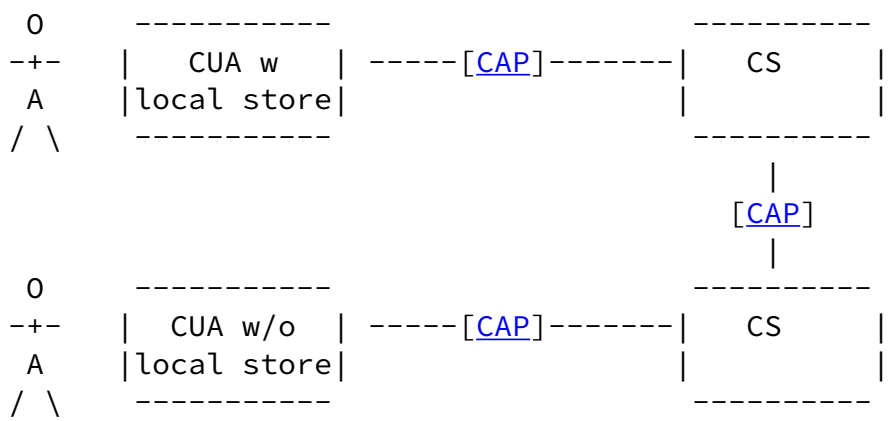
3.2.5 Users communicating on a multi-user system

Users on a multi-user system may schedule meetings with each other using [CAP]-enabled CUA and service. The CUA may or may not have a local store. If they do, then they need to ensure that the data on the CUA is synchronized with the data on the CS.



3.2.6 Users communicating through different multi-user systems

Users on a multi-user system may need to schedule meetings with user on a different multi user system. The services can communicate using [CAP] or [ITIP].



4. Important Aspects

There are a number of important aspects of these calendaring documents that people, especially implementors, should be aware of.

4.1 Timezones

The dates and times in components can refer to timezones. These timezones can be defined in some central store, or they may be defined by a user to fit his or her needs. Any user and application should be aware of timezones and timezone differences.

4.2 Choice of Transport

There are issues to be aware of in choosing a transport mechanism. The choices are a network protocol, such as CAP, or a store and forward (email) solution.

The use of a network ("on-the-wire") mechanism may require some organizations to make provisions to allow calendaring traffic to traverse a corporate firewall on the required ports. Depending on the organizational culture, this may be a challenging social exercise.

The use of an email-based mechanism exposes innately time-sensitive data to unbounded latency. Large or heavily utilized mail systems may experience an unacceptable delay in message receipt.

4.3 Security

See the "Security Considerations" section below.

4.4 Amount of data

In some cases a component may be very large. For instance, some attachments may be very large. Some applications may be low-bandwidth or be limited in the amount of data they can store. The size of the data may be controlled in [CAP], by specifying maximums. In [iMIP] it can be controlled, by restricting the maximum size of the email that the application can download.

4.5 Recurring Components

In [iCAL] one can specify complex recurrence rules for VEVENTs, VTODOs, and VJOURNALs. There is the danger that applications interpret these rules differently. Thus, one must make sure that one is careful with recurrence rules.

5. Open Issues

Many issues are not currently resolved by these protocols, and many desirable features are not yet provided. Some of the more prominent ones follow.

5.1 Scheduling people, not calendars

Meetings are scheduled with people, however people may have many calendars, and may store these calendars in many places. There may also be many routes to contact them. These protocols do not attempt to provide unique access for contacting a single person. Instead, 'calendar addresses' are booked, which may be email addresses or individual calendars. It is up to the users themselves to orchestrate mechanisms to ensure that the bookings go to the right place.

5.2 Administration

These protocols do not address the issues of administering users and calendars on a calendar service. This must be handled by proprietary mechanisms for each implementation.

5.3 Notification

People often wish to be notified of upcoming events, new events, or

changes to events. These protocols do not attempt to address these needs in a real-time fashion. Instead, the ability to store alarm information on events is provided, which can be used to provide client-side notification of upcoming events. To organize notification of new or changed events clients will have to poll the data store.

6. Security considerations

6.1 Access Control

There has to be reasonable granularity in the configuration options for access to data through [CAP], so that what should be released to requestors is, and what shouldn't isn't. Details of handling this are described in [CAP].

6.2 Authentication

Access control must be coupled with a good authentication system, so that the right people get the right information. For [CAP] this means requiring authentication before any data base access can be performed, and checking access rights and authentication credentials before releasing information. [CAP] uses SASL for this authentication. In [IMIP], this may present some challenges, as authentication is often not a consideration in store-and-forward protocols.

Authentication is also important for scheduling, in that receivers of scheduling messages should be able to validate the apparent sender. Since scheduling messages are wrapped in MIME, signing and encryption is available for free. For messages transmitted over mail this is the only available alternative. It is suggested that developers take care in implementing the security features in [IMIP], bearing in mind that the concept and need may be foreign or non-obvious to users, yet essential for the system to function as they might expect.

The real-time protocols provide for the authentication of users, and the preservation of that authentication information, allowing for validation by the receiving end-user or server.

6.3 Using email

Because scheduling information can be transmitted over mail without any authentication information, email spoofing is extremely easy if the receiver is not checking for authentication. It is suggested that implementors consider requiring authentication as a default, using mechanisms such as are described in Section 2 of [IMIP].

The use of email, and the potential for anonymous connections, means that 'calendar spam' is possible. Developers should consider this threat when designing systems, particularly those that allow for automated request processing.

6.4 Other issues

The current security context should be obvious to users. Because the underlying mechanisms may not be clear to users, efforts to make clear the current state in the UI should be made. One example is the 'lock' icon used in some web browsers during secure connections.

With both [IMIP] and [CAP], the possibilities of Denial of Service attacks must be considered. The ability to flood a calendar system with bogus requests is likely to be exploited once these systems become widely deployed, and detection and recovery methods will need to be considered.

7. Acknowledgements

Thanks to the following who have participated in the development of this document:

Eric Busboom, Pat Egen, David Madeo, Shawn Packwood, Bruce Kahn.

8. Bibliography

[ICAL] [RFC-2445] Calendaring and Scheduling Core Object Specification
[ITIP] [RFC-2446] iCalendar Transport-Independent Interoperability Protocol
[IMIP] [RFC-2447] iCalendar Message-Based Interoperability Protocol
[IRIP] draft-ietf-calsch-irip iCalendar Real-time Interoperability Protocol
[CAP] draft-ietf-calsch-cap Calendar Access Protocol

[RFC-1847] Security Multiparts for MIME
[RFC-2045] MIME Part 1: Format of Internet Message Bodies
[RFC-2046] MIME Part 2: Media Types
[RFC 2047] MIME Part 3: Message Header Extensions for Non-ASCII Text
[RFC-2048] MIME Part 4: Registration Procedures
[RFC-2049] MIME Part 5: Conformance Criteria and Examples

9. Author's Addresses

Alexander Taler
Email: alex@elea.dhs.org

Bob Mahoney
MIT
E40-327
77 Massachusetts Avenue
Cambridge, MA 02139
Tel: (617) 253-0774
Email: bobmah@mit.edu

George Babics
Steltor (formerly CS&T/Lexacom)
2000 Peel Street
Montreal, Quebec, Canada
H3A 2W5
Tel: (514) 733-8500 x4201
Fax: (514) 733-8878
mailto: georgeb@steltor.com

10. Full Copyright Statement