TITLE: **Calendar Interoperability Testing Report – January 2006 Provo Utah**
Version: 1.0
Date of Publication: February 9, 2006

Contributing Authors:

Patricia Egen, David Nuttell, Dave Camp, Joey Minta, Grant Baillie, Simon Vaillancourt, Chuck Norris, Dan Markham, Dan Hickman, Mike Douglass,

STATEMENT OF INTELLECTUAL PROPERTY RIGHTS

STATEMENT OF APPROPRIATE USAGE

DISCLAIMER OF WARRANTY

TABLE OF CONTENTS

CALENDAR INTEROPERABILITY EVENT
January 2006
Provo, Utah

Introduction

The following document contains notes and results from the January 2006 calendar interoperability event held at the Novell complex in Provo, Utah.

The chart below shows the attendees, their organization and the products they were testing.

Attendees:

Attendees	Organization	Products
Chuck Norris	EVDB	CALDAV server
Danny Markham	EVDB	CALDAV server
Joey Minta	Mozilla	Mozilla client
Dave Camp	Novell	Hula, Evolution
Dave Nuttell	Novell	Groupwise
Simon Vaillancourt	Oracle	Oracle CALDAV server
Grant Bailey	OSAF	Chandler and Cosmo
Mike Douglass	RPI	RPI CALDAV Server
Dan Hickman	Trumba	OneCalendar

General Comments

After introductions there were general discussions about testing and requirements for suites. The Open Source tool, iCal 4J, does have a parser so we are considering pointing people at already-developed implementations for testing icalendar streams.

We talked about the concept of using a public CALDAV server to help test calendar objects. This server and suite would need to do the following tasks: Import events, do some actions, export and compare. It would then present the final state to the tester. A good parser will test and decompile into components.

The purpose of this test event was to test CALDAV clients and servers and iCalendar implementations. Exhibit A are the test scenarios used for iCalendar testing. All products tested during this event are "not shipping application." Therefore, in this pubic document, we do not explicitly name products or note how each product works. Comments are general and generic. To help differentiate notes, we have named the participants as Vendor one, Vendor two, Vendor three, Vendor four, Vendor five, Vendor six and Vendor 7.

Vendor Notes and Comments

The following are various notes, including comments provided by vendors that reflect their findings and thoughts from the interoperability event.

The CALDAV server implementations had the easiest task at the event. Their job was to ensure the servers were available and to monitor the traffic from the various clients. As clients sent requests, various bugs were fixed and added to list of items that need to be repaired.

Several clients went through the CALDAV scenarios (details are attached to this report). Each time we hold the interoperability events, we uncover more bugs that each client and server repairs.

One thing noted was a difference in CALDAV specifications within products. The spec is now at version 8; however, some of the other clients and/or server were still only at version 7. This caused some things to not be able to be tested at this event. However, this is common when a draft is not a finalized RFC.

Some servers imported timezones well. Recurrence is not working yet for some of the clients. This shows how two vendors have two different interpretations of the same object.

One vendor had a few problems with timezone ids. They appeared to be too verbose for their product. There were little issues with starts and ends. The vendor said it was helpful to hear other issues.

Not every client supports RFC 2446 or 2447. These are the scheduling and email transport protocols. In general, it seemed that the majority of clients did not have much support for attendees, and as such, a vast portion of the testing scenarios were not applicable.

Vendor one noted they lack implementations of some of the more complex RRULE (like BySETPOS), and struggled with overridden instances of recurring events. Publishing and subscribing seemed to work quite well with all clients. Their CalDAV support is rather limited at this stage, but they were able to do basic event adding/deleting with all servers except for one. That vendor was running a different version of the spec, and as such, could not return their queries.

Vendor two reported the following items:

- They currently support import and export of RFC 2445 iCalendar objects and also allow clients to publish and subscribe to calendars via Webdav.
- Calendar objects from Vendor three were successfully published and subscribed to their calendars via Webdav. Webdav connectivity worked quite well.
- They noted that they were not able to map Vendor one's TZID's to their timezones.
- In this case, timezones were interpreted in UTC.
- Their floating dates and all day events imported correctly into another vendor's product.
- They also successfully mapped the other vendor's timezones to their internal timezones.
- Their application did not correctly import private event status (CLASS:PRIVATE). This has since been fixed.
- Vendor four noted that importing their ics object into Vendor two's application worked well. Currently Vendor four does not support recurrence or timezones.

- Importing one time event ics objects from Vendor five worked correctly. The sample recurring event instance provided from Vendor five included multiple DTSTART properties which is not valid and could not be imported. Importing their recurring events into Vendor five's application worked for the most part. Import of a rescheduled occurrence of a recurring event left the original date in addition to the new date/time in Vendor five's application.
- When importing Vendor six's ics files, timezone ID's were not correctly mapped to their timezones. They had to implement enhanced mapping logic. Imported recurrence samples worked very well with all rescheduled events importing properly. Export of Vendor six's client recurring event did not import correctly. The master event included only RDATE objects to specify recurrence set and did not include an RRULE which is not currently supported by Vendor two.
- Vendor seven provided four sample ics files which all successfully imported. Examples included multiple timezones, all day events and recurrence. Vendor two exports include a custom attribute in the VTIMEZONE component which caused issues when importing into Vendor seven's application. These should be ignored.

Vendor seven reported the following items:

- A couple of bugs relating to Vendor eight's handling of recurring events and EXDATEs.
- An issue where they were PROPPATCH'ing calendar collections (to set their display names). However, not all servers support this, and they need to ignore returned errors in this case.
- A couple of Vendor five problems that were fixed, allowing tests to proceed.
- A issue where they do not correctly interpret an empty multistatus response to a PROPFIND (Vendor five does this, and it's allowed by the WebDAV spec), leading to a failure in creating calendar collections.
- Updating events on Vendor six is broken right now, apparently because of a known issue with Vendor seven not preserving X-properties.
- They are unable to parse .ics files from Vendor two - (a bug in the way we parse VTIMEZONE).
- Some of the other vendors noticed that they are exporting events in UTC timezone incorrectly.
- A couple of bugs in their subscription to http: ics URLs were uncovered.
- There were initially some problems with Vendor five's client that turned out to be a result of client and server having implemented different versions of the CalDAV spec.
- Vendor five updated their calendar-query REPORTs to match the latest spec, but was unable to retrieve events from our application. This was tracked down to being an indexing issue, and was fixed. Subsequently, the testing ran OK.
- After the above fix, Vendor three had similar behavior retrieving events. It turned out this was a result of PUT-ing illegal ICalendar data (an empty STATUS value in a VEVENT). Vendor three worked around the problem, and was able to proceed. However, our application should have rejected the initial PUT request: This problem is being investigated.

Vendor five reported the following items:

- Vendor five more interested in finding out what worked and what didn't than in actually following the test scenarios. So the matrix was filled in with what seemed to work and where problems were found.
- Most of the other vendors didn't actually send invitations but rather sent emails with the ICS file attachments that were then imported. Therefore there was not a lot of Accept and Decline testing done. Delegate, Counter and Reschedule was not tested with any vendors.
- They felt it was very helpful and they are busy working on the problems they found.
- They noted that their CALDAV implementation has problems with "@" in URLs. It also isn't happy with PUT not returning an ETAG.
- Their CALDAV application had problems with Vendor six's Auto-added organizer.
- Vendor eight doesn't add a "calendar-access" DAV header.

Summary

Once again, this was a good event. We tried a new approach for testing – instead of trying to work through any bugs, we decided to continue testing all items that we could and only go back to fix bugs if they were holding up continuing with testing. This turned out to be a good approach and we were able to get the majority of vendor products tested.

Location turned out to be an important issue during the interop. Location appears to be useless in iCalendar. Some clients use locations, some do not. There needs to be a definition of properties that are absolutely required. Mozilla commented that they drop the location details on their recurrence items. Everyone wanted to be able to ingest location items and then know what to do with them. There may need to be extensions put in place within the specifications or via the Calsify efforts to handle them. Xprops are what need to be enhanced/fixed/resolved.

For the next interop, items to add to interop testing include:

- how many people use the language property on icalendar objects
- how many can support daily savings time changes that will happen in 2007
- Free busy testing within CALDAV
- Tasks testing
- More testing of RFC2446 and RFC2447 scheduling events
- and the next phase of the CALDAV specification.

Following this report are the test results for the scenario testing for both CALDAV and the iCalendar specifications.

My thanks to everyone who furnished their notes and results.

Respectfully submitted,

Pat Egen
Interoperability Event Manager

Exhibit A - iCalendar Test Scenarios

Basic test scenarios:

A: Non-repeating cases:
- 1: User A PUBLISHes an event
- 2: User A invites Users B, C, D & E to a meeting:
  - A: ATTACHments:
    - 1: 0
    - 2: 1
    - 3: 1+
  - B: ALTREPs of:
    - 1: DESCRIPTION
    - 2: COMMENT
    - 3: CONTACT
    - 4: LOCATION
    - 5: RESOURCES
    - 6: SUMMARY
    - 7: iana-token (TBD usage but legal)
    - 8: x-token (TBD usage but legal)
  - C: Including ALARMs:
    - 1: AUDIO only
    - 2: DISPLAY only
    - 3: EMAIL only
    - 4: PROCEDURE only
    - 5: iana-token only (TBD value but non X- type)
    - 6: x-token only (TBD value but anything made up is ok)
    - 7+: Multiple alarm types (mix & match 1-6 above as desired)
  - D: COMMENTs:
    - 1: 0
    - 2: 1
    - 3: 1+
  - E: CONTACTs:
    - 1: 0
    - 2: 1
    - 3: 1+
  - F: ATTENDEE property parameters:
    - 1: CUTYPE:
      - A: INDIVIDUAL (Default)
      - B: GROUP
      - C: RESOURCE
      - D: ROOM
      - E: UNKNOWN
      - F: x-name (TBD case, perhaps SKiCAL?)
      - G: iana-token (TBD case)
      - H: Multiple values (Illegal case)

2: MEMBER
   A: 0
   B: 1
   C: 1+
3: ROLE:
   A: CHAIR
   B: REQ-PARTICIPANT (Default)
   C: OPT-PARTICIPANT
   D: NON-PARTICIPANT
   E: x-name (TBD case, perhaps SKiCAL?)
   F: iana-token
   G: Multiple values (Illegal case)
4: PARTSTAT:
   A: NEEDS-ACTION (Default)
   B: ACCEPTED
   C: DECLINED
   D: TENTATIVE
   E: DELEGATED
   F: x-name (TBD case, perhaps SKiCAL?)
   G: iana-token
   H: COMPLETED (Illegal for VEVENTs)
   I: IN-PROCESS (Illegal for VEVENTs)
   J: Multiple values (Illegal case)
5: RSVP
   A: TRUE
   B: FALSE (Default)
   C: Any other value (Illegal case)
   D: Multiple values (Illegal case)
6: DELEGATED-TO
   A: 0
   B: 1
   C: 1+
7: DELEGATED-FROM
   A: 0
   B: 1
   C: 1+
8: SENT-BY
   A: 0
   B: 1
   C: 1+ (Illegal case)
9: CN
   A: 0
   B: 1
   C: 1+ (Illegal case)
10: DIR
   A: 0

B: 1
C: 1+ (Illegal case)

3: User B Accepts the invitation:
A: but then Declines the invitation:
B: but then requests a Refresh of the invitation:

4: User C Declines the invitation:
A: but then Accepts the invitation:
B: but then requests a Refresh of the invitation:

5: User D Counters with a new meeting time:
A: User A Declines the Counter:
B: User A Accepts the Counter and reschedules the meeting:

6: User E Delegates to User G:
A: User G Accepts the invitation:
B: User G Declines the invitation:
C: User G requests a Refresh of the invitation:
D: User G Counters with a new meeting time:
E: User G Delegates to User I:

7: User A reschedules the meeting:
Repeat permutations of 1-6 below here as necessary.

B: Repeating cases:
(Repeat A. subcases but expand for instance manipulation including entire set, 1 instance, THISANDPRIOR & THISANDFUTURE ranges
Tests should include the following permutations:
RDATEs only
RRULEs only
RDATEs and RRULEs
RDATEs & EXDATEs only
RRULEs & EXDATEs only
RDATEs & EXRULEs only
RRULEs & EXRULEs only
RDATEs, EXDATEs & EXRULEs
RRULEs, EXDATEs & EXRULEs
RDATEs, RRULEs & EXDATEs
RDATEs, RRULEs & EXRULEs
RDATEs, RRULEs, EXDATEs & EXRULEs

Exhibit B - CALDAV Testing Scenarios

1.	**Event creation.**
1.1.	Create new single-instance meeting titled "Meeting 1.1" with the location "Durham".
1.2.	Create new meeting titled "Meeting 1.2" recurring every Monday from 10:00 AM to 11:00 AM for 4 weeks.
1.3.	Create new single-instance meeting titled "Meeting 1.3" with 2 other attendees.
1.4.	Create new single-instance meeting titled "Meeting 1.4" with an alarm set to trigger 15 minutes prior to the schedule time of the meeting.
2.	**Event modification**
2.1.	Modify the title of meeting "Meeting 1.1" to "Meeting 1.1bis".
2.2.	Modify the location of the meeting "Meeting 1.1bis" to "Seattle bis".
2.3.	Reschedule meeting "Meeting 1.1bis" to the next day.
2.4.	Add an attendee to "Meeting 1.1bis".
2.5.	Add an alarm to "Meeting 1.1bis".
2.6.	Modify the title of the 1st instance of the recurring meeting created in 1.2.
2.7.	Modify the participation status of the 1st attendee in meeting 1.3 to DECLINED.
2.8.	Cancel the 4th instance of the recurring meeting created in 1.2.
2.9.	One client changes "Meeting 1.1bis" to a different time, second client 'refreshes' its display to see the modification.
3.	**Event retrieval**
3.1.	calendar-query REPORT
3.1.1.	No filtering (match everything)
3.1.1.1.	Query all components and return all data. (tests <calendar-query> and <filter>)
3.1.1.2.	Query all components and return ETag WebDAV property and all data. (tests <calendar-query>+ <DAV:prop> and <filter>)
3.1.1.3.	Query all components and return just entire VEVENT components. (tests <calendar-query> , <filter>+<comp-filter>)
3.1.1.4.	Query all components and return VEVENT components with only DTSTART, DTEND/DURATION, SUMMARY, UID, SEQUENCE, RRULE, RDATE, EXRULE, EXDATE, RECURRENCE-ID. (tests <calendar-query> , <filter>+<comp-filter>, <calendar-data>+<comp>+<prop>)
3.1.2.	time-range filtering
3.1.2.1.	Query all components within a one day time-range and return all data. Make sure that there is a recurring event that starts prior to the chosen time-range but has one non-overridden instance within the time-range. (tests <calendar-query> , <filter>+<time-range>)
3.1.2.2.	Query all components within a one week time-range and return just entire VEVENT components. Make sure that there is a recurring event that starts prior to the chosen time-range but has one overridden instance within the time-range. (tests <calendar-query> , <filter>+<time-range>)
3.1.3.	component based filtering
3.1.3.1.	Query all components that contain an embedded VALARM component. (tests <calendar-query> , <filter>+<comp-filter>)

3.1.3.2.	Query all components that contain an embedded VALARM component whose trigger falls within a specific time-range. (tests <calendar-query> , <filter>+<comp-filter>+<prop-filter>+<time-range>)
3.1.4.	property based filtering
3.1.4.1.	Query all components that contain any ORGANIZER property. (tests <calendar-query> , <filter>+<prop-filter>+<is-defined>)
3.1.4.2.	Query all components that contain an ORGANIZER property with a specific CUA text value case-insensitively. (tests <calendar-query> , <filter>+<prop-filter>+<text-match>+<caseless>)
3.1.4.3.	Query all components that contain an ORGANIZER property with a specific CUA text value case-senstively. (tests <calendar-query> , <filter>+<prop-filter>+<text-match>+<caseless>)
3.1.5.	parameter based filtering
3.1.5.1.	Query all components that contain a DTSTART property with a TZID parameter. (tests <calendar-query> , <filter>+<prop-filter>+<text-match>+<param-filter>+<is-defined>)
3.1.5.2.	Query all components that contain an ATTENDEE property with PARTSTAT=NEEDS-ACTION parameter. (tests <calendar-query> , <filter>+<prop-filter>+<text-match>+<param-filter>+<text-match>)
3.2.	calendar-multiget REPORT
3.2.1.	Query a specific href and return all data. (tests <calendar-multiget >)
3.2.2.	Query multiple hrefs (some of which do not exist) and return all data. (tests <calendar-multiget >)
3.2.3.	Query a specific href and return ETag WebDAV property and all data. (tests <calendar-multiget >+ <DAV:prop >)
3.2.4.	Query multiple hrefs (some of which do not exist) and return ETag WebDAV property and all data. (tests <calendar-multiget >+ <DAV:prop >)
3.2.5.	Query a specific href and return VEVENT components with only DTSTART, DTEND/DURATION, SUMMARY, UID, SEQUENCE, RRULE, RDATE, EXRULE, EXDATE, RECURRENCE-ID. (tests <calendar-query >, <calendar-data>+<comp>+<prop>)
3.2.6.	Query multiple hrefs (some of which do not exist) and return VEVENT components with only DTSTART, DTEND/DURATION, SUMMARY, UID, SEQUENCE, RRULE, RDATE, EXRULE, EXDATE, RECURRENCE-ID. (tests <calendar-query >, <calendar-data>+<comp>+<prop>)
4.	**Event deletion**
4.1.	Delete a single non-recurring meeting.
4.2.	Delete a single recurring meeting with no overridden instances.
4.3.	Delete a single recurring meeting with overridden instances.
4.4.	Delete a non-overridden instance of a recurring meeting.
4.5.	Delete an overridden instance of a recurring meeting.
5.	**Access Control**
5.1.	View access control details on current user's main calendar.
5.2.	Change access control details on current user's main calendar to add another user with read-only access. Verify that other user can view the calendar but not change it.

5.3.	Change access control details on current user's main calendar to add another user with read-write access. Verify that other user can view the calendar and change it. Verify that changes done by one user are seen by the other.
5.4.	Remove another user's access to the current user's main calendar and verify they can no longer access the calendar.
6	**Calendar Management**
6.1	Browse the list of calendars on the server, including the current user's personal calendars.
6.2	Create a new calendar in the current user's personal calendar space.
6.3	Create a regular collection in the current user's personal calendar space.
6.4	Create a new calendar inside the collection created in 6.3.
6.5	Delete the calendar created in 6.2.
6.6	Delete the collection created in 6.3.