

# Upcoming Slony-I Enhancements

Christopher Browne  
Afilias Canada



# Windows Support

- Presently, Slony-I is not supported on Windows™
- Some users have verified that the triggers and stored functions can work on Windows™ but slon must run on Unix
- A team plans to integrate Windows support into Slony-I; likely coming in version 1.2.
- They plan to make pgAdmin usable as a GUI alternative to slonik

# Improved Test Bed

- Current “duct tape” tests depend on X11, and require manual operation
- Work is under way on a more portable and automatic testing framework
- This should make it easier to run regression tests and more regularly verify functionality on more platforms

# Performance Enhancement - COPY\_SET

- Drop/recreate indices during COPY\_SET can let subscriptions run several times faster

# Performance Enhancement – Event Handling

Presently, event propagation leads to a lot of dead tuples in `pg_listener`

Alternatives include:

- Grouping responses together to diminish numbers of events
- Plans ongoing to have `pg_listener` stored in-memory in future PG release
- Use Spread (part of planned Slony-II infrastructure) to propagate events
- Slon does too many notifies... Nodes should keep polling for events...

# Performance Enhancement – Sequences

- At present, having 500 sequences means their values are all collected in each SYNC
- Most are likely changed infrequently
- Having tables that are infrequently updated is very cheap as entries only collect in `sl_log_n` when updates take place
- Work ongoing to find a way to diminish the DB work for sequences

# Custom Events

Slony-I supports a certain set of events; an interesting extension would be to introduce your own custom events.

Application raises an event:

```
select 0LTP.createevent('BILL_TXN',  
    'I742AL', 'USD', '27.95', now());
```

A custom slon node then sits on the network listening for these events.

# Custom Filters

- The notion here is that a filter would be applied to the primary key and the set of updates
- A subscriber might take “filtered” results, so it only considers updates for US activity, or European activity, or such
- This would require parsing queries, and filtering COPY\_SET; definitely non-trivial

# Uses for Log Shipping

When input comes into a “log shipping” node, off line, numerous modifications are possible:

- Use triggers/rules to automatically generate/update materialized views
- Use triggers/rules so that stateful updates are transformed into temporal updates, maintaining a temporal database

# Lagged Nodes

- Being asynchronous, Slony-I could support having nodes held back by chosen periods of time
- It would be natural to add a slon option to “lag” by some time interval
- Need a parameter indicating “stop at SYNC #X – patch already in CVS HEAD
- Alternatively, have slon “lag” rather than terminating