Interlisp-D AR Fields

Interlisp-D software support and development uses an "Action Request" data base system for keeping track of bug reports and requests for features. We take feature requests as serious as bug reports -- don't hesitate to ask. Users are encouraged to submit ARs, either via mail, electronic mail, or (for internal Xerox users) directly using the AREDIT facility within Interlisp-D. The following documents the fields we use in ARs and what they mean:

AR identification

Number: Every AR has a number, which increases by one for each AR submitted. AR numbers are *never* recycled, and ARs are *never* deleted. The AR number is automatically generated.

Source: This is an arbitrary string which should contain the name and electronic address of original customer or internal Xerox users reporting problem. Customers should be identified by "Liason (Customer name)", e.g., Raim.pasa (Bill White @ Teknowledge). This field is used in sending mail back to find out more technical details, or to notify people about the change in status. If multiple people report the same bug, each one should be included in the Source field.

Problem Description

Subject: A terse summary of problem, enough to identify it uniquely. "FOO doesn't work" or "Floppy problem" is not good enough. Think of yourself as a newspaper headline writer: "Attempt to write file when Floppy door open causes awful noise". Implementors may change the Subject field as more details about the true nature of the problem becomes apparent. Feature requests generally start with "Want", e.g., "Want way to make windows triangular rather than square."

Problem Type: What kind of problem report or feature request:

Bug	The system does not work as documented.
Implementation	The system works, but the internal structure is wrong. (Generally submitted by other implementors or looking at the sources.)
Interface	The design of the user interface is wrong. Includes problems in the way in which things display, as well as program callable structures.
Feature Request	Request for a new feature or set of facilities
Documentation	The system works, but the documentation is wrong, unclear, or incomplete.
Performance	The system works, but it is too slow doing the described operation.

Description: This field should contain the complete description of problem or request, including any subsequent discussion. If bug reports come in via electronic mail, put the whole message in this field. Edit relevant info into the beginning of the Description, especially if it summarizes what the problem really is.

Frequency: How reproducable is the problem? Leave blank if irrelevant (e.g., feature request.) Generally only relevant for bug reports. One of:

Everytime	reproducible every time.
Intermittent	doesn't always happen.
Once	saw it happen once.

Impact: How seriously does it affect your ability to get work done, value of Interlisp-D, etc. The names apply to bug reports, but feature requests should be rated along analogous lines.

Fatal	causes system crash, loss of work, etc. requirement for project completion.
Serious	can be worked around but seriously interferes with work, requires substantial reimplementation
Moderate	tolerable, but clearly a problem, responsibility of Interlisp development
Annoying	annoying problem, minor request for new feature that "would be

nice"

Minor — may be some dispute about whether it is even a bug, very minor feature request.

Test Case: This field isn't used for what the name might imply: it should be a list of the files needed to recreate problem. Recipe for reproducing the problem (which is what you might think a Test Case was for) should be in the Description.

Lisp Version: This should be either the release name (Fugue.1, Carol, Harmony) and MAKESYSDATE in which the problem occurs (or the feature doesn't occur.) The Lisp AREDIT package attempts to fill this in; if you submit from a different system that you are running in, please change it. If its a documentation problem, include the date of the documentation.

Machine: What machine does problem occur on (one of 1108, 1132, 1100). Leave blank if *all*.

Disk: What kind of disk is on the machine? (only fill in if relevant to AR.) Constrained to have the known disk types for Machine.

Microcode Version: (automatically generated, delete if known to be irrelevant, e.g., for documentation problems.)

Memory Size: (automatically generated. Delete if known to be irrelevant.)

File Server: If problem deals with communication, what server are you talking to? (This field should really have "Server" on it, rather than File Server.) One of VAX/VMS, VAX/UNIX, 8037, etc.

Server Software Version: As appropriate for the server you're talking to.

System: **Subsystem**: Category & sub-category of problem type. Subject to change. System includes: Lisp Software, System Software, Text and Graphics, Documentation, IO Architecture. Generally these are filled in by LispSupport, as it corresponds more to our own internal project structure.

Problem disposition

Workaround: If relevant, this field can contain a known procedure to work around problem until it is fixed; generaly a short recipe belongs here.

Status: Status of AR as it moves thru the system:

New — All ARs start out as New.

Open — Has been looked at by LispSupport; all fields are filled in & has been assigned.

Fixed — problem fixed, in LispCore loadup. The In/By: field is set to the next release name. The Assigned-to: field is set.

Closed — System with fix in it has been tested & released.

Declined — Request for feature officially *never* going to be implemented (e.g., we think its a bad idea). Bug report considered spurious (we don't think it is a bug)

Superseded — Another AR includes the problem described in this one. In this case, the Subject of this AR should include the AR# of the one that supercedes it, and the superceding AR should contain the union of information in this one.

Obsolete — e.g., module in which problem reported is no longer supported.

Incomplete — The information submitted is not enough to take action -- not enough information to identify the bug, or the feature request doesn't spell out in enough detail what is wanted. This is different from Declined in that the request is considered valid, but open for more detail.

In/By: What version of Interlisp-D has/will have this problem fixed? (E.g., Harmony, Intermezzo, Jazz, Fugue.6, etc.)

Disposition: Brief notes explaining changes to status, plus automatically generated description of who changed status when.

Difficulty: Rough estimate filled in by developer on scope of problem.

Easy	< 1 week to fix
Moderate	< 1 month to fix
Hard	< 6 months to fix
Very Hard	> 6 months to fix
Impossible	can't be fixed
Design	requires a design

Priority: Development's estimate of whether it will be in the "next" release. (Changes from release to release).

Absolutely	Release will be held if not completed.
Hopefully	Major release goal. Schedule slip admitted, but likely to get completed.
Perhaps	Will get implemented if other revisions in same area are completed.
Unlikely	Unlikely to be included in the next release.

Assigned To: The developer who 'took care' of this AR, either by fixing it or declining it. Currently only after AR is fixed.

Attn: The developer(s) who should look at this AR. We're moving away from relying on this.

Submitter: The user name (and registry) of the person who actually did the Submit. Automatically generated by AR submit, not editable.

Date: The date the AR was originally submitted. Automatically generated by AR submit, not editable.

Edit-By: person who edit this AR last. Automatically generated by the Lisp AR edit tool. (The Disposition field contains more info about edits history.)

Edit-Date: date when this AR was last edited. Also automatically generated.