Journal Of Harmonized Research (JOHR)

Journal Of Harmonized Research in Applied Sciences
4(2), 2016, 90-92

ISSN 2321 – 7456

Original Research Article

# CREATING AN ENVIRONMENT FOR REUSABLE SOFTWARE RESEARCH

**Alireza Rezaee[1] and Mazyar Pajohesh[2]**

[1]Assistant Professor of Department of System and Mechatronics Engineering, Faculty of New Sciences and Technologies, University of Tehran, Tehran, Iran
[2]Master of student of Adiban University, Garmsar, Semnan, Iran

**Abstract:** When a change to the architecture used to develop environmental assessment RESOLVE / C ++, it became clear to rewrite the environment from the beginning. At the start of the principles of software reuse it again, leading to a significant reduction in code, increases in maintenance and bug fixes were possible.

**Keywords:** Reusability, RESOLVE, C++, Lisp, XEmacs, GNU Emacs.

**Introduction:** Resolve a software engineering framework, language, and discipline in the design, specification, construction and use of focused parts. [6] The importance of field, composition, understanding, and performance. Our principles to resolve / C ++ [2] [7] [3] [1] [4] application development environment and reap big rewards.

**Background:** This reusable software management group (RSRG) in computer and information sciences at Ohio State University (OSU-CIS) is done on a Unix-based environment. Tools used include makeup, different language compilers, and GNU Emac. For our purposes, CS1 and CS2 environment of students taking and using RESOLVE / C ++ is essentially the same.

In 1998, OSU- CIS Great Migration from UNIX to another one began. In doing so, our many years of baggage left behind, including some ancient applications, window managers, and the like. At the same time, it was decided to migrate to GNU EmacsXEmacs be done. Which was available in the old GNU Emacs, XEmacs is available in the new.

RESOLVE / C ++ and settlement / ADA development and maintenance of the old environment with a locally customized version of Ada, C ++, and state lock that is part of GNU Emacs was supported font. A simple test showed that the old Emacs Lisp RESOLVE / C ++ under XEmacs does not work, we have to consider this code and functionality are presented.

**Elements:** Since RESOLVE / Ada is also actively looking to do, we were able to limit our scope to deal with the determination / C ++ respectively. Old code to support this

**For Correspondence:**
arrezaee@ut.ac.ir
Received on: June 2016
Accepted after revision: June 2016
Downloaded from: www.johronline.com

environment S-2468 Emacs Lisp expression spread across five files had been formed. These files include `` lift 'version of the C ++ - mode and font lock between 1996 and 1998. Elements of the package of settlement / C ++ was incorporated into different files stored locally were. The number of problems is provided.

**Stagnation:** Since the code a lift version of another package, it is practical to incorporate bug fixes, performance improvements, feature enhancements, etc., that occurred during the year. The code was written in 1996 remained almost exactly as it was.

**Maintainability:** Maintenance of the code suffers a great deal. It was very difficult, even for LISP programmer skills, read the source code and easily understands it. Made numerous dependencies need to open multiple files at once to trace functionality mode from beginning to end. Since 1996 only updates the built environment in the form of language keywords were added or deleted.

**Fragility:** Changes to the Code strange tendency to affect the rest of the system was unpredictable way. Sometimes, parts of the code simply stop working, for unknown reasons. In practice, proved to be working normally up blocks of code.

**Exclusivity:** Because the determination / C ++ environment, a modified version of `` normal 'C ++ environment, it was impossible to use both RESOLVE / C ++ and `` vanilla' 'C ++. The determination of the code / C ++ in a significant path to be able to use the `` natural " 'C ++ environment.

**New Version:** Requirements for settlement / C ++ environment was clear upon examination of the state of the system.

- Must be easily understood, especially as Lisp's expertise scarce. Non experts should be able to understand the code.
- Effective should do. With the heavy use of thin clients, and supports remote users, many instances of the software can be running on a single processor. Users should not believe that your environment `` Pokey " is.
- Should be adjustable. The default settings for things like color must be connected to your state.
- Local should not require maintenance. Due to budget constraints, it is possible to maintain a system to keep lisp guru. The nature of open source software is dynamic, where new features are added and bugs are fixed [5] as XEmacs and its C ++ - support for LISP packages to be updated, the changes need to resolve and will / C ++ environment will be incorporated. The area where the settlement only acceptable changes / C ++ language and the environment need to slowly change your keyword list will be updated. Those keywords should be in one, easy to maintain the list. Given these conditions, a number of decisions were reached.
- Instead of hardwiring color and font settings to the environment, we parameter settings, put them in a file that researchers and students new options .xemacs RSRG or your classes can inherit. This enables the user to customize these settings without having XEmacs through the tuning process twice (once, at the start of the case, and again when evaluating user preferences set) the process.
- A higher version of available modes is unacceptable.
- Major mode to support RESOLVE / C ++ is derived `` state " of CC- case, a package which is used to support the C, C ++, Objective C, Java, CORBA IDL, and code Pike.

The implementation of a new major mode to support RESOLVE / C ++ is. The result is a single source file 125 S-. Our needs have been met, and we have seen the determination of the new XEmacs / C ++ environment through a major upgrade (XEmacs 20.4 to 21.1p2) live. A new platform (XEmacs 21.1p2 in Win32), was tested and working mode as expected. It is now possible to choose whether to use RESOLVE / C ++ (rcpp mode) or standard C ++ (C ++ - mode).

**Conclusions:** Rcpp successfully reusable software model that does not work outside the lab.

With the construction of a component through clearly defined interfaces and mechanisms were using their environment, We are able to offer the same functionality with about 5% of the code needed to support similar functionality using a fork and `` custom '' respectively. The gains in speed, storage, and portability are.

**References**

**[1]** P. Bucci, T.J. Long, and B.W. Weide. Teaching software architecture principles in CS1/CS2. In Proceedings 3rd International Software Architecture Workshop, pages 9-12. ACM, November 1998.

**[2]**T.J. Long and B.W. WeideWeaving software engineering into the fabric of CS1 and CS2. In Proceedings 4th International Workshop on Software Engineering Education, pages 66-69, May 1997.

**[3]**T.J. Long, B.W. Weide, P. Bucci, D.S. Gibson, J.E. Hollingsworth, M. Sitaraman, and S.H. Edwards. Providing intellectual focus to CS1/CS2. In Proceedings 29th SIGCSE Technical Symposium on Computer Science Education, pages 252-256. ACM, February 1998.

[4] T.J. Long, B.W. Weide, P. Bucci, and M. Sitaraman. Client view first: An exodus from implementation-biased teaching. In Proceedings 30th SIGCSE Technical Symposium on Computer Science Education, pages 136-140. ACM, March 1999.

[5] Eric S. Raymond. The Cathedral and the Bazaar, July 1999. [online]http://www.tuxedo.org/~esr/writings/cathedral-bazaar/cathedral-bazaar.htm l.

[6]M.curtin, Creating an Environment for reusable software research: A case study in Reusability,OSU-CISRC-89/99-TR21,1999.

[7]M. Sitaraman, B.W. Weide, T.J. Long, and W.D. Heym. Teaching the essential role of mathematical modeling in understanding and reasoning about objects. Technical Report OSU-CISRC-9/97-TR43, Department of Computer and Information Science, The Ohio State University, September 1997.