The CodeTime Certification Strategy

by K. Sean Halle

University of California at Santa Cruz

Email: seanhalle@yahoo.com

Web: codetime.sourceforge.net

Sept 14, 2006

Abstract

This paper presents the proposed strategy for enforcing a uniform adoption of the CodeTime platform. We suggest the use of dual-licensing to support a non-profit "steward" of the platform. The steward uses revenues for three main purposes: 1) Coordinate development of the platform, 2) Perform conformance testing and 3) Actively defend the use of the CodeTime trademark and branding.

Open source software is maturing. We draw upon the experiences of Sun with defending the Java brand, of MySQL's commercial viability with a dual licensing model, PHP's stewarding foundation, and Linux's wide acceptance.

1 Introduction

We have the following mechanisms for the indicated reasons:

1. Dual licensing – produces a reliable revenue stream to support the stewardship functions
2. Project management – wide-scale adoption needs an organised, coordinated effort
3. Documentation – reduces the barrier to use, thereby increasing adoption rate
4. Services – training and consulting reduce the barrier to entry for hardware and software providers
5. Conformance testing – increases reliability, coherence, quality and enables robust defense of brand
6. Legal defense of brand – prevents take-over by proprietary "extensions" and related attacks
7. Promotion of brand – increases rate of adoption and licensing revenues, and weakens forks

1.1 Dependencies

Several circular dependencies interlock.

Licensing revenues depend upon having a strong, defensible brand. A strong defensible brand depends upon marketing and promotion efforts, and depends upon legal efforts to find infringements and take action to stop them. These in turn depend upon having large enough licensing revenues to support those activities.

Having wide adoption depends upon having a reliable product that is easily obtainable, easy to get started with, and that can be depended upon to work with the software and hardware that claim to work with it. These things in turn depend upon having a coherent development process, adequate testing, good documentation, enforced conformance testing, and a defended brand. All of these in turn depend upon a revenue stream to support them, which in turn depends upon having wide acceptance to attract enough paying licensees.

1.2 Lessons learned from other Open Source endeavers

Lessons from Linux: The adoption rate of Linux has saturated in part because of the proliferation of distributions; a software package that claims to work on Linux often only works on one or a few of the distributions. We wish to prevent this kind of dis-harmony. Our conformance testing must be passed by a third-party vendor before they can place our trademark on their product, or make any claims of compatibility in their literature. This will ensure that every component bearing our trademark works reliably with other components bearing our trademark (assuming our conformance process works properly, see section ⟨reference|⟩).

Lessons from Sun: It is possible to prevent proprietary extensions from locking users in to a proprietary platform. We wish to follow Sun's legal practices in defending our brand. We believe that their highly public win against Microsoft helped foster trust within the developer community which in turn helped to promote adoption of Java. We also wish to follow their marketing model in promoting the adoption of our platform. We also wish to follow Sun's successful strategy of insisting on single-source control of the development process. We believe this was necessary to achieve the reliability of Java releases and has also been instrumental in Java's wide acceptance.

Lessons from MySQL: Dual licensing does work. GPL has a wide recognition and valuable good-will, therefore we plan to release the platform under GPL. We also plan to follow MySQL's legal strategy for its commercial licensing. Finally, Red Hat, MySQL, Sleepy Cat and others have shown that services for open source software are desired and represent a significant revenue stream to help support development and brand defense efforts.

Lessons from PHP: Stewardship foundations do work. Setting up a non-profit is an acceptable method to the developer community. The non-profit status and support mission appease fears of "making money off MY efforts" that many in the developer community have towards purely commercial software.

1.3 Conformance Testing

The conformance suite uses the reference test-harness. It runs a suite of reference programs on the reference Virtual Server in the reference test harness. Each program comes with a test suite that includes the expected results. The program is run inside the test harness, presented with each test case and the results are checked by the test harness.

The conformance suite includes 3rd party commercial software. To be included, a company must pay a fee to have their software be a standard. This fee will cover the increased hardware and maintenance costs of testing with that additional software package.

Hardware manufacturers will pay a fee to have their Virtual Server implementation tested for conformance. This is done by un-plugging the reference implementation and plugging in the hardware manufacturers implementation, then running the conformance suite.

Development tools, compilers, distribution-packagers, and so on are tested against the defined interface for that platform-element. The foundation defines an interface for each component, all interaction with that component from other elements of the platform can only go through the standard interface mechanism and must conform to it (see legal protections).

A program cannot receive certification without providing a test suite plus expected results in the format of the standard test harness. A program cannot be marketed using the trademark without being certified.

1.4 Legal Protections

The right to use the trademark comes with legal restrictions. The requirements and the consequential actions are stated in the contract each comany signs when they receive the right to use the trademark. The consequence, spelled out in the contract, for losing the trademark is: upon notice from the foundation, the company must notify by certified mail all registered customers who have purchased the product that it has lost certification and include a statement from the foundation giving the reason for the loss of certification in language of the foundation's choosing. In addition, all packages must be pulled from distribution and sales channel shelves. And all promotional materials must be pulled.

If a program provides functionality that was not tested in the test suite, then it loses certification at the point the foundation becomes aware of this.

If a language implementation includes features that were not disclosed in written documentation and pointed out in the required summary of features, or, if there is any way for the comiler of that language to produce a compiled image whose intermediate format differs in any way from the intermediate format defined by the Foundation, then that language implementation loses certification at the time the foundation becomes aware of the difference. (Prevents Microsoft from hijacking with "custom" features in their compiler plus "custom" features in their Virtual Server).

If a Virtual Server accepts an intermediate format that differs in any way from the one defined by the Foundation, that VS loses certification. (IE, no "additional" features that are "improvements").

Any protocol embedded within the performance-info section of the intermediate format must be released as public domain, free of any restrictions. Further, any distributed image which uses this protocol must perform reasonably, where reasonably is defined by the Foundation alone, on Virtual Servers which do not recognise the protocol, and the exact same results must be obtained whether the protocol is recognised and used or not. (The intermediate format has a section which is unstructured. It is intended for performance information, in whatever format is dreamt up. This paragraph's legal clause prevents this section from being used as an end-run around conformance. It disallows crippling the normal implementation, and disallows using the performance-info section as an extension to the rest of the intermediate format, and makes the protocols used in the section public and free for any party to use).

If a development tool does not implement (and/or did not test in the required test suite) a required interface, or provides extensions to the interface (say, for "performance" reasons), they lose certification.

Every element of the platform has a defined interface. No extensions to that interface are allowed. Every portion of that interface must be implemented. The entire interface must be effectively tested in the test suite provided by the manufacturer. This requires that any component can be unplugged and a replacement from any third party can be plugged in instead with no difference in communication between the components. This ensures coherent inter-operation, when each component is effectively tested against the same interface.

In practice, the Foundation will be very forgiving. It wants to promote the use of the platform and can only do that with the good will of companies participating in the platform. These protections are in place to protect the coherence of the platform and will only be used when repeated behavior of a company over time threatens the value of the trademark, and therefore the value to all the other companies that participate in the platform, and the value to all the customers who have paid with either time and effort or dollars for components of the platform like software.

1.5 Foundation Structure

The foundation will be a non-profit 501c3 company. It will have an evolving structure the way most companies do, from startup with only a few people, through maturity. It is envisioned that the foundation will grow large as the platform becomes the industry standard for parallel software.

The following major functions are envisaged:

1. Industry relations – interfacing with commercial entities who are interested in or use the platform. This includes marketing efforts, contract negotiations, and requests for contract services. Works closely with legal when potential violations of contracts come to light. The main point of negotiations to correct violations.

2. Services – provides experts who help companies implement their products for the platform. Most critical is a pool of Virtual Server implementation expertise that hardware manufacturers can contract with to develop or help with development of a Virtual Server for their hardware.

3. Development Management – organises the development activities for the platform. In particular, interfaces with any volunteer developers who participate via the open source license of the platform elements.

4. Development – performs in-house development of platform elements, defines the specifications of interfaces and functionality of platform components.

5. QA – performs testing of the platform elements as part of the development process. Tests both foundation-driven development and open-source-driven development.
6. Conformance – separate from QA, this function develops and maintains the conformance testing process. It maintains the machines which run the conformance tests, performs the tests themselves, and maintains the test suites, adding them in to the automated testing process.
7. Documentation – proofs the documentation received through the open source development process, interviews the developers, and writes and extends the documentation. Also writes the documentation for the foundation-driven development, interviewing developers. It also receives requests for improvements through the public bug reporting and request process, and modifies documentation and the documentation process accordingly.
8. Legal – writes contracts, patents intellectual property, maintains copyright. It also looks for violations of trademark and contracts and enforces the consequences when negotiations through Relations break down.

1.6 Conclusion

We have proposed a system we believe will effectively ensure coherent development of the CodeTime platform. The system involves a not-for-profit stewarship entity that defines standards, tests conformance to that standard, and pursues breaches of the standard.