JOURNAL OF OBJECT TECHNOLOGY

Online at http://www.jot.fm. Published by ETH Zurich, Chair of Software Engineering ©**JOT, 2004**

Vol. 3, No. 5, May-June 2004

Open Source Reusable Assets

Mahesh H. Dodani, IBM Software, U.S.A.

1 OPEN SOURCE EVERYWHERE

Open source has become the de facto community process for the Internet. The success of this process has primarily been in software development, starting with its roots in Linux (http://www.linux.org/), and has spread to hundreds of diverse projects ranging from designing a better intravenous system for third world patients (http://www.thinkcycle.org/), improving cooking recipes (http://www.ibiblio.org/oscookbook/), solving crime (http://doenetwork.org/), calculating Pi (http://projectpi.sourceforge.net/), and writing textbooks for academia (http://otp.inlimine.org/.)

In software engineering, open source projects are profilerating in software development. Open source software development is very compelling since it nurtures the evolution of the software by allowing programmers to read, redistribute, and modify the source code for the software. Developers are motivated to fix bugs in the software and improve or adapt it. Since the development effort and work is opened to a potentially large group of developers, the evolution can happen at an astonishing speed especially compared to the slow pace of conventional software development.

The open source community has learned that this rapid evolutionary process produces better software than the traditional closed model, in which only a very few programmers can see the source.

Besides operating systems, commercial open source projects can be found in all major areas of software engineering, for example:

- Personal information management applications: The Open Source Application Foundation (http://www.osafoundation.org/) is working on developing an application that combines email, calendering, file-sharing and instant messaging that can be easily changed by users.
- Application development tools: Eclipse (http://www.eclipse.org/) is an open platform for the integration of tools within an integrated development environment. Eclipse provides tool developers flexibility and control of their software technology by allowing them to integrate their tool and technology on

top of a platform that operates under a open source common public license that provides royalty free source code and world wide redistribution rights.

- HTTP Server: The Apache HTTP server (http://httpd.apache.org/) is probably the most commercially used open sourced application on two-thirds of the web servers on the Internet to serve up web pages and other data. This open-source HTTP server is available for both UNIX and Windows operating systems and provides a secure, efficient and extensible server that provides services that conform to HTTP standards.

Each of these open source projects has the following common objectives and characteristics, which revolve around community sharing:

- Goals of the project shared by the community. This is the key requirement for the success of any open source project – to get the community to buy-in to a shared need and and the approach of the project to meet these needs.
- Work on the project shared by the community. Every successful open project has a thriving community of participants who share the work that is needed to keep the project evolving. This objective poses a requirement that the project work can be broken up into smaller chunks that can be worked on by interest participants, and that these chunks can then be integrated to achieve the project goal.
- Results of the project shared by the community. Finally, the community must be able to easily access and use the results of the project. Note, this does not imply that the results are free – the community must be able to use the results in an "open" way allowing for redistribution, change, access to source code, etc. This access to the results is usually through some form of General Public License agreements that has been established by the open source community (see http://www.opensource.org/licenses/ for a list of open source licenses.)

2 REUSABLE ASSET COMPONENTS AND LIFECYCLE

Now let's turn our attention to reusable assets. As I have reported in several earlier columns, tackling the complex needs of on demand businesses (e.g. http://www.jot.fm/issues/issue_2004_01/column7) requires large reusable assets that can support and guide the design of reasonable solutions. To be effective, these reusable assets can include the following components as summarized in Figure 1:

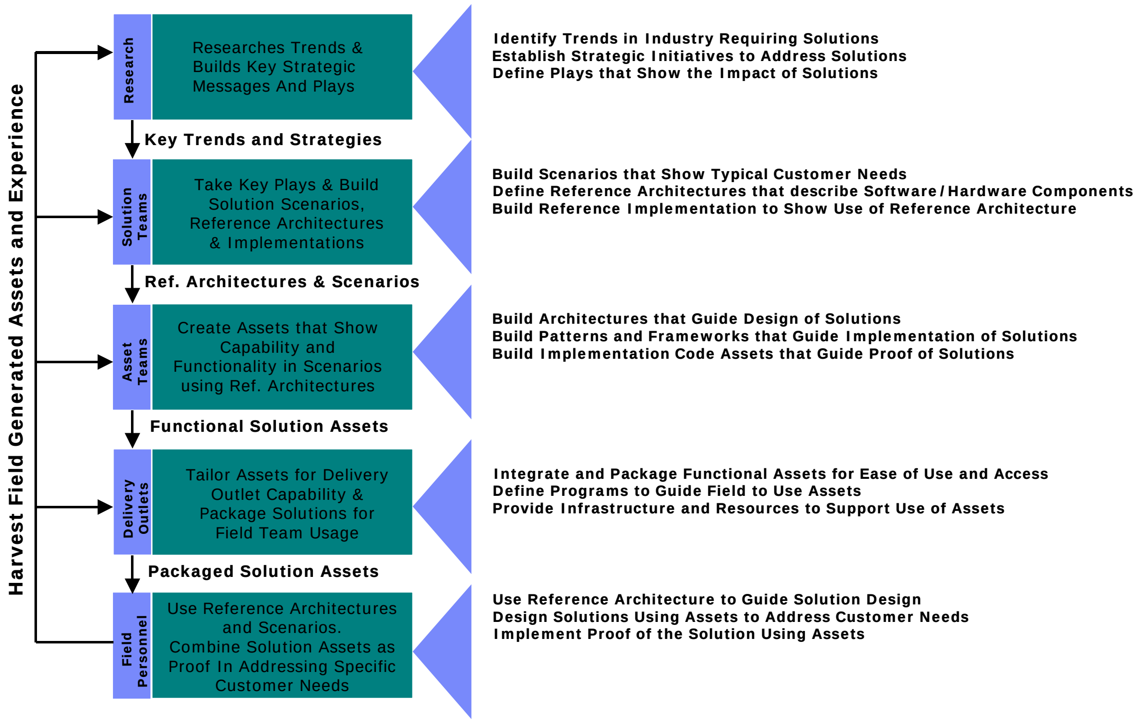

Harvest Field Generated Assets and Experience
Research
Researches Trends & Builds Key Strategic Messages And Plays
Identify Trends in Industry Requiring Solutions
Establish Strategic Initiatives to Address Solutions
Define Plays that Show the Impact of Solutions
Key Trends and Strategies
Solution Teams
Take Key Plays & Build Solution Scenarios, Reference Architectures & Implementations
Build Scenarios that Show Typical Customer Needs
Define Reference Architectures that describe Software/Hardware Components
Build Reference Implementation to Show Use of Reference Architecture
Ref. Architectures & Scenarios
Asset Teams
Create Assets that Show Capability and Functionality in Scenarios using Ref. Architectures
Build Architectures that Guide Design of Solutions
Build Patterns and Frameworks that Guide Implementation of Solutions
Build Implementation Code Assets that Guide Proof of Solutions
Functional Solution Assets
Delivery Outlets
Tailor Assets for Delivery Outlet Capability & Package Solutions for Field Team Usage
Integrate and Package Functional Assets for Ease of Use and Access
Define Programs to Guide Field to Use Assets
Provide Infrastructure and Resources to Support Use of Assets
Packaged Solution Assets
Field Personnel
Use Reference Architectures and Scenarios. Combine Solution Assets as Proof In Addressing Specific Customer Needs
Use Reference Architecture to Guide Solution Design
Design Solutions Using Assets to Address Customer Needs
Implement Proof of the Solution Using Assets

Figure 1: Reusable Asset Components and LifeCycle

- Key Trends and Strategies: This component describes the key trends in the field that are driving the needs of companies to which the set of assets can provide solutions, and outline the strategic initiatives that can drive these solutions to address customer needs. The primary use of these components is to ensure that the solution and corresponding assets match the needs of the customer.
- Reference Architectures and Solution Scenarios: This component provides the hardware and software components that are needed to build end-to-end solutions that are described by the scenarios. These reference architectures provide the blueprint for the solutions, and the scenarios define the typical customer needs that these solutions can be applied.
- Functional Solution Assets: This component provides a set of proven, tested assets that can drive the design, development, implementation and extension of solutions. This component can range from architectural artifacts that support and guide the design of solutions to implementation code artifacts that can be used to show a proof of the designed solution. An example of a functional architectural asset is the IBM Patterns for e-business (http://www-106.ibm.com/developerworks/patterns/.) They match business challenges with Business and Integration patterns, use proven Application and Runtime patterns, populate the Runtime patterns with pre-tested Runtime Product Mappings, and establish best practice guidelines for application design, development and management.

- Packaged Solution Assets: The complexity of the solution requires many asset artifacts, and in order for the field personnel to use these effectively they must be packaged together. Furthermore, there is a need to support the use of these assets through delivery outlets (that can provide hardware/software infrastructure and supporting resources), and programs that guide the field to use the assets effectively with their customers.

A key observation of the asset components and lifecycle shown in Figure 1 is that different communities produce each of the asset components (and in the case of the functional solution assets, different communities produce the artifacts), and that the feedback loop of taking advantage of field experience to evolve the various components is typically an input from the field to the appropriate community. This situation usually leads to the following major problems with current approaches to complex reusable assets:

- Disconnected communities lead to disconnected assets. Since each community defines its own standards, templates and processes for developing, distributing, and evolving their asset, it becomes very difficult to integrate the assets into a comprehensive package that can be effectively used by the field in addressing customer needs.
- Evolution of components at different rates leads to difficulties in making assets work together. The evolution of each component is dependent on the community that is responsible for the component, and therefore the rate of change depends entirely on the established priorities and processes. This leads to components that evolve at different rates, so that at any given point in time the components are out of sync with each other.
- People who (re)use assets are not involved in making changes to evolve the asset. The field personnel are the primary users of the assets and therefore have the experience on how the asset needs to evolve to better handle the ever-changing customer requirements. However, typically this experience is provided to the community in charge of the components and the established priorities and processes determine how these experiences will be used in evolving the asset components.

Can open source approaches be used to alleviate these problems?

3 OPEN SOURCE REUSABLE ASSETS

Figure 2 shows the overall design of applying an open source model to handling reusable assets components that are built, used and modified by several communities of practitioners.

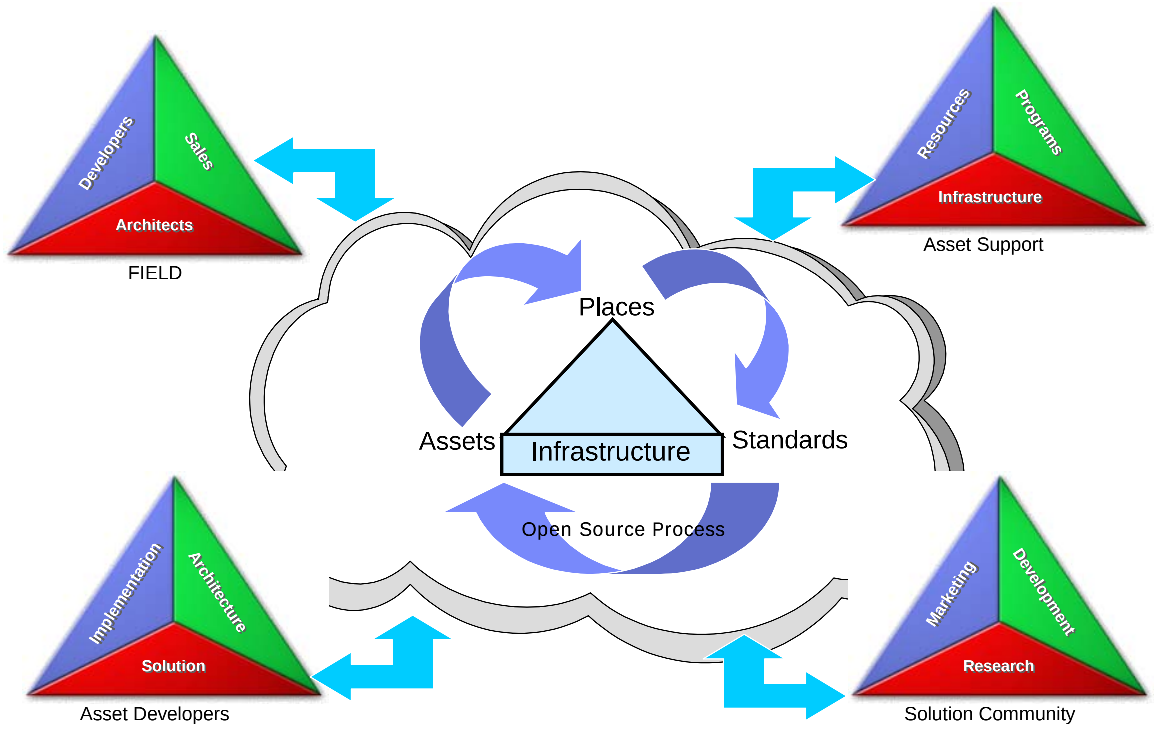


Figure 2: Open Source Reusable Assets

Providing an open source collaborative environments for the reusable assets communities of interest to thrive require at least the following infrastructure and support as shown in the Figure:

- Places are portals where practitioners meet, collaborate, access information, and access applications. The community portal should link in with learning and knowledge portals to ensure consistency of ideas and approaches. A governance model must be established to ensure that the community places are coordinated, integrated and consistent.
- Assets include the entire range of components that are needed to make the asset usable and effective, including reference architectures, patterns, reference implementations, code, architecture and design work products, benchmarking and performance demonstrations, case studies, etc. These asset components need to be organized and managed by a configuration and version control system to facilitate check-in/check-out and versioning. The environment needs to support regular “builds” for release to communities
- Standards provide the guidelines and approach to ensure that the communities can work together. The environment supports a set of “standard” software and hardware configurations to ensure that asset components can be integrated effectively. There are common components to support sharing including work product definitions and templates. There is a need for an appropriate infrastructure to support the community including distributed repositories, support for peer-to-

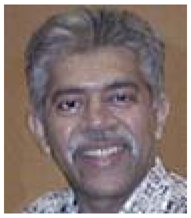
peer collaboration, information integration capabilities, and remote access to assets.

- The open source process ensures that the community of practitioners has a shared goal, can share the work effort needed to evolve the assets, and can reuse the assets effectively. To facilitate communities of interest around asset components, the following approach would be effective:
  - The primary community of interest provides the initial “seed” for the asset component.
  - Any member of the community can modify the asset component by checking out the asset component, making changes to the component, testing the changes, and recommending the change to the governing body.
  - A group of community leaders defined for each asset component is defined to nurture the evolution of the asset component. These community leaders have responsibilities to monitor changes, evaluate proposals, accept changes, and ensure currency of the component.

In conclusion, managing the development, use and evolution of complex reusable assets requires many communities of interest to share a common goal, share the work effort, and share the results. These objectives and ideals are the very tenets of the open source model. Applying an open source approach to reusable assets can ensure that they stay current, focused, and effective.

Open Sesame! Enter the open source environment, and enjoy the fruits of your labor.

About the author



Mahesh Dodani is an e-business architect with IBM Software. His primary interests are in enabling communities of practitioners to design and build complex e-business solutions. He can be reached at dodani@us.ibm.com.