Promote-2.5-Tests & Quizzes

See the Comment section at the bottom of this page for discussion on this tool's promotion.

Core Criteria Matrix

The table below is derived from the Criteria for Supported Status document and is intended as a quick-reference guide on how this tool measures up to the requirements for a Core tool.

Key:
✅ - Criteria met.
⚠ - Work needs to be done to meet criteria.
❌ - Will not meet criteria by code freeze.
N/A - Not Applicable.

Community Support Criteria	**Tests & Quizzes (a.k.a. Samigo)**
There must be an identified group of committers within the Sakai community who take responsibility for the tool. This group can be comprised of the original author(s) or others from the community willing to take responsibility for the application.	✅ (Stanford & others TBD)
The application source code must reside in Sakai's source control system. Organization and management of the source control system is defined in a different Sakai Community Practice (to be determined).	✅
At least two Sakai production sites must be utilizing the tool successfully in their production environments; please list the sites. Those sites should be able to report that the tool runs properly and is useful to their users. These sites must also report that the application does not cause any problems with other parts of the Sakai release such as poor performance, memory leaks, etc.	✅ (Univ. of Cape Town, Claremont Univ. Consortium, Arizona State Univ. /IDEAL, Stanford, all...)
The developer(s) of the tool must be committed to maintaining the application across Sakai version changes including any necessary data conversion for their elements of Sakai if storage structure changes.	✅
The developer(s) of the tool must be willing to answer questions about their application on Sakai dev list.	✅
The developer(s) of the tool must commit to helping to develop test plans and specifications for the application. Until those test plans are in place, the author(s) will commit to join the Sakai release/QA process and perform the necessary QA on their application. To become an official tool, the author(s) of the tool must contribute test plans and specifications (to satisfy the requirements of the QA group).	✅ (current set)
Bugs and feature requests for the tool must be tracked in Sakai's bug tracking system.	✅
Technical Elements Criteria	**Tests & Quizzes (a.k.a. Samigo)**
If the tool persists data to a database, it must support all official Sakai databases (currently HSQL, MySql, and Oracle).	✅
The tool must work properly with the Sakai AutoDDL approach. The application must properly create tables when tables do not exist and the tables must be named so as not to conflict with other Sakai tables.	✅
All database access to tables that the tool does not own will take place via published Sakai APIs provided by the application that manages those tables. The tool should use APIs for access to its own data, but it must use APIs for access to data from others.	✅
The tool must participate in the system-wide configuration (sakai.properties) and not require any local configuration to be hand-edited.	✅
The tool must properly operate in the Sakai Authorization and tool placement structure. It must either use existing appropriate security functions or introduce new security functions for the application.	✅
The tool will not require patches to other Sakai tools or to the Sakai framework. A application may require changes to other areas of Sakai, but those changes should be negotiated ahead of time and should be part of the full distribution.	✅
The tool must fully work in a clustered Sakai application server environment.	✅
The tool cannot force new jars into shared/lib or common/lib. Sakai's goal is to keep these jar footprints as small as possible. Any new jar requirement in those areas requires significant discussion.	✅
There are a number of system-wide elements in Sakai including Spring, Hibernate, and others--the application must work with the versions of these elements that are part of the Sakai release.	✅
Licensing must be clean before the application is moved into Sakai's source control system and put into a Sakai distribution. The license must not violate the policies set by the Sakai Foundation. This also means that a tool cannot require other application or jar with a proprietary or ECL-incompatible license to be part of the distribution.	✅
Interaction and Visual Design Criteria	**Tests & Quizzes (a.k.a. Samigo)**

The tool UI must look like the rest of the Sakai application and properly inherit skins from Sakai (i.e. when the sites color changes the tool colors change as well). The tool should not look "out of place" amongst other Sakai tools.	✅
The tool should follow general interaction guidelines outlined in the Sakai Style Guide (SG) so users have consistent experiences and expectations about how to complete actions such as "paging in a list", "navigating between pages", "taking action of items in a list", etc across tools. The SG is a guide and should be used to help inform interaction decisions but is not meant to have all the answers.	✅
UI components available in the Sakai library should be used where possible (e.g. wysiwyg editor, calendar, paging widget).	✅
The application must support all of the browsers currently supported by Sakai including FireFox/Mozilla and Internet Explorer.	✅
The tool should provide basic help which integrates seamlessly with the Sakai Help system. The help should have a look and feel and writing style similar to the other elements of help.	✅
Desirable Elements Criteria	**Tests & Quizzes (a.k.a. Samigo)**
The tool is properly internationalized with all of its user-displayed strings derived from properties files.	✅
The tool should be fully accessible and pass a Sakai accessibility review.	✅ (needs a new review, last review was 2.0)
If the tool has any persistence or business rule code it should be factored into a Sakai Component with a published API which has complete javaDoc.	⚠️ (javadoc can be improved)
This component should be designed so as to allow other applications and/or system processes to be able to work with the business objects handled by the application.	✅
Creation of web services API for the tool. The authors should provide a web-service layer which sits on top of their API.	❌ (did create proof-of-concept for alti-lab demo in 2005)
The tool should participate in the Sakai cross-site import and export if it persists business objects.	✅
The tool should be inter-operable with other Sakai tools where appropriate.	✅ (currently gradebook, and site import/export)
The tool should generate event codes that are triggered minimally on new, revise, and delete actions on the basic objects created by the tool.	⚠️ (filed as SAK-11137)
Hibernate is not required but if the tool is doing significant ORM, it should be done using Hibernate - the application should work with the current version of Hibernate used in the Sakai release.	✅
In order for Sakai to seen to be a seamless application, effort needs to be made to minimize overlap in functionality with other Sakai "bundled" tools. At times, as tools are evolving, or complete replacement tools are being produced, overlap is acceptable. Where overlap will happen between tools under consideration, developers should first look to fixing/improving the existing tools rather than simply writing a new tool with a few features. Each tool which adds overlap may result in the long term need to "clean up" the overlap to maintain a good user experience.	✅
Notification and Process Criteria	**Tests & Quizzes (a.k.a. Samigo)**
30-day notification prior to code freeze of the following	
What it is	✅ homepage
How to install/configure it	✅ 2.4 release notes
Known issues	✅ see release notes
Whom to contact	✅ see homepage