Component Test Plan

SBAC-11Test Authoring

Version 1.7

Revision History

Date	Version	Description	Author
02/06/14	1.0	Initial Draft and introduction	Ryan Marinello
02/07/14	1.1	Requirement tracing	Ryan Marinello
02/11/14	1.2	Format and template changes	Ryan Marinello
02/14/14	1.3	Level 2's	Ryan Marinello
02/19/14	1.4	API	Ryan Marinello
02/21/14	1.5	Version of Source document tracking	Ryan Marinello
02/24/14	1.6	API Revised	Ryan Marinello
02/25/14	1.7	Dependent Services	Ryan Marinello
03/18/14	1.8	Test Spec Bank L2 's interfacing	Ryan Marinello
03/21/14	1.9	Validation changes	Ryan Marinello
03/19/14	2.0	Publishing changes	Ryan Marinello

Contents

I. Purpose

The purpose of this document is to describe the testing strategy and test scenarios applied to the tested component.

I. Overview

1. The overall responsibilities of this component are:
    a. Creation and management of Assessments
    b. Creation and management of Subjects
    c. Creation and management of Publications
    d. Creation and management of Scoring Functions
    e. Creation and management of Item Selection Algorithms
    f. Content Management of Test Specification Data
    g. Creation and management of Blueprints
    h. Interaction with Test Item Bank for item retrieval
    i. Creation and management of Test Packages
    j. Interaction with Test Specification Bank
    k. Item pools and Item Groups
    l. Test forms (adaptive and fixed)
    m. SSO and Tenancy chain
    n. Integration with Core standards web services

The user audience for Test Authoring component is:

1) Functionally diverse with integration of tenancy relationships and role based permissions. The various users are responsible for creation, management and approval of assessments and assessment related data.

II. Component: Test Authoring

The Test Author component is responsible for the creation, management and workflow of test assessment specification data. Test Authoring integrates with the Test Specification Bank component for storage of published assessment data, retrieval of data and integrates with Test Item Bank for item data based on item metadata, item pools and test forms. During our testing we integrated AIR's Core Standards web services for subjects, publications, have been integrated to provide test data for standards publications with information feeding assessment blueprints and other uses. Test Authoring's final integration is with Test Packager, which is a downstream component of Test Authoring.

III. Scope of Testing

This plan focuses on verification and validation testing of the Test Authoring component based on the Level 2 component requirements. Mission statement: prove level 2 requirements for Test Authoring which are believed to be correct. Heuristic: CBC (contextual, beneficial, congruent)

1. Verification Testing

Verification testing will cover the functional testing of requirements.

Requirement Document	Document Name	Location	Version
Level I Requirements	DRC SBAC11 Requirements – Test Authoring Level 1	Knowledge Tree -> High Level Requirements -> DRC	V.5
Level II Requirements	DRC SBAC11 Requirements – Test Authoring Level II	Knowledge Tree -> Requirements -> Level II Requirements	V.1

2. Validation Testing

Validation testing will cover scenario testing.

Scenario	Endpoints and examples
Display and Describe the Test Authoring API RESTful Services and Controllers	1. http://localhost:8080/rest/api 2. /api/blueprintElement 3. /api/item 4. /api/subject 5. /api/progman 6. /api/form 7. /api/user 8. /api/coreStandard 9. /api/publication 10. /api/itemSelectionAlgorithm 11. /api/blueprintSock 12. /api/segment 13. /api/itemGroup 14. /api/scoringRule 15. /api/assessment 16. /api/scoringFunction 17. /api/reportingMeasure 18. /api/enemy 19. /api/tibitem 20. /api/publishingRecord
RESTful Response Codes	1. 200 for GET and PUT with response content 2. 201 for POST with response content 3. 204 for DELETE no response message 4. 404 for Not supported endpoint 5. 500 Error(s) from un-supported requests
http://localhost:8080/rest/api/blueprintElement Content-Type: application/json Supported Methods: GET, PUT	/blueprintElement GET / 200 codes /blueprintElement/Id PUT /blueprintElement/Id/segment/Id/add PUT /blueprintElement/Id/synch PUT /blueprintElement/Id/segment/Id/remove PUT

	/blueprintElement/Id/activate PUT /blueprintElement/Id/fillRemaining PUT /blueprintElement/Id/clear PUT
http://localhost:8080/rest/api/item Content-Type: application/JSON Supported Methods: GET,POST,DELETE	/item GET and POST and PUT /item/Id GET and PUT and DELETE *integrated Test Item Bank endpoint
http://localhost:8080/rest/api/subject Content-Type: application/JSON Supported Methods: GET, PUT, POST	/subject GET and POST /subject/Id PUT /subject/delete/Id PUT *integrated core standards web service endpoint
http://localhost:8080/rest/api/progman Content-Type: application/JSON Supported Methods: GET, PUT, POST	/progman/teanant/component/name/Id GET /progman/tenantTypes GET /progman/tenant/Id GET
http://localhost:8080/rest/api/form Content-Type: application/JSON Supported Methods: GET,POST, DELETE	/form GET and POST /form/Id GET and PUT and DELETE
http://localhost:8080/rest/user Content-Type: application/JSON Supported Methods: GET	/user/tenantchain GET /user/assets GET
http://localhost:8080/rest/coreStandard Content-Type: application/JSON Supported Methods: GET	/coreStandard /coreStandard/publication/CC-ELA-v1/grade GET /coreStandard/publication/CC-ELA-v1/standard GET *integrated core standards web service endpoint

http://localhost:8080/rest/publication Content-Type: Application/JSON Supported Methods: GET, POST, PUT, DELETE	/publication GET and POST /publication/Id GET and PUT and DELETE *integrated core standards web service endpoint
http://localhost:8080/rest/itemSelectionAlgorithm Content-Type: Application/JSON Supported Methods: GET,PUT, POST,DELETE	/itemSelectionAlgorithm GET and POST /itemSelectionAlgorithm/Id GET and PUT and DELETE /itemSelectionAlgorithm/itemSelectionPurposes GET /itemSelectionAlgorithm/itemSelectionTypes GET
http://localhost:8080/rest/blueprintSock Content-Type: Application/JSON Supported Methods: GET, POST, PUT, DELETE	/blueprintSock GET and POST /blueprintSock/Id GET and PUT and DELETE /blueprint/Id/synch PUT
http://localhost:8080/rest/segment Content-Type: Application/JSON Supported Methods: GET,POST,PUT,DELETE	/segment GET and POST /segment/Id GET and PUT
http://localhost:8080/rest/itemGroup Content-Type: Application/JSON Supported Methods: GET, POST, PUT, DELETE	/itemGroup GET and POST /itemGroup/Id PUT and DELETE
http://localhost:8080/rest/scoringRule Content-Type: Application/JSON Supported Methods: GET,POST, PUT, DELETE	/scoringRule GET, PUT, POST /scoringRule/Id GET, PUT, DELETE

http://localhost:8080/rest/assessment Content-Type: Application/JSON Supported Methods GET, POST, PUT, DELETE	/assessment GET and POST /assessment/Id GET and PUT /assessment/Id/validateblueprint GET, DELETE /assessment/type GET
http://localhost:8080/rest/scoringFunction Content-Type: Application/JSON Supported Method GET,POST,PUT,DELETE	/scoringFunction GET, POST, DELETE /scoringFunction/Id PUT /scoringFunction/computationRuleTypes GET /scoringFunction/computationlRuleMultiplicityTypes GET /scoringFunction/dictionaryIndexTypes GET
http://localhost:8080/rest/enemy Content-Type: Application/JSON Supported Methods GET, POST, PUT, DELETE	/enemy GET and POST /enemy/Id GET and PUT
http://localhost:8080/rest/tibitem Content-Type: Application/JSON Supported Methods GET	/tibitem GET /tibitem/apipitem_TEST GET /tibitem?version=1.1 GET or other metadata fields
http://localhost:8080/rest/publishingRecord Content-Type: Application/JSON Supported Methods: GET, PUT	/publishingRecord GET /publishingRecord/id GET and PUT /publishingRecord/specificationTypes /publishingRecord/specificationTypes/id GET /publishingRecord/publishingStatusTypes GET

Tenancy Chain example
Create /rest/tenant
Content-Type: application/json
	Create Tenant by endpoint. Method: POST
	http://localhost:8080/rest/tenant
	{
	“name” : “Minnesota”,
	“type”:”STATE”
	}
	Response Code: 201
	{
	"id" : "5249d53be4b0d4f99546dc19",

	"name" : "Minnesota", "description" : null, "type" : "STATE", "tenantSubscriptions" : null, "url" : "/tenant/5249d53be4b0d4f99546dc19" }

Tenancy Chain example
Update /rest/tenant
Content-Type: application/json
	Update Tenant information by endpoint. Method: PUT
	http://localhost:8080/rest/tenant/{id}
	{
	"id" : "5249d53be4b0d4f99546dc19",
	"name" : "Wisconsin",
	"type" : "STATE"
	}
	
	Response Code: 200
	{
	"id" : "5249d53be4b0d4f99546dc16",
	"name" : "Wisconsin",
	"description" : null,
	"type" : "STATE",
	"tenantSubscriptions" : null,
	"url" : "/tenant/5249d53be4b0d4f99546dc19"
	}

Tenancy Chain example
Search /rest/tenant /{id}
Content-Type: application/json
	Query String parameters for Tenant Method: GET
	http://localhost:8080/rest/tenant/{id}
	Response Code: 200
	{
	"id" : "5249d53be4b0d4f99546dc16",
	"name" : "Wisconsin",
	"description" : null,
	"type" : "STATE",
	"tenantSubscriptions" : null,
	"url" : "/tenant/5249d53be4b0d4f99546dc19"
	}

DOMAIN	Assessments, Blueprints, Computational Rules, Conversions, Core Standards, Enemy, Forms, Items, Reporting, Publishing, Scoring, Segments, Subjects and TIB
SEARCH	Assessments, Blueprints, Enemy, Forms, Items, Publications, Publishing, Reporting, Scoring, Segments, Subject
VALIDATIONS	Item Selection, Scoring Functions, Scoring Rules
PUBLISH	Registration, Simulation, Administration, Scoring, Complete

Describe the Services	Dependencies
http	AssessmentService
http	BlueprintElementService
http	BlueprintSockService
http	CoreStandardsService
http	CurrentUserService
http	EnemyService
http	FormService
http	ItemGroupService
http	ItemSelectionAlgorithmService
http	ItemService
http	NavigationService
http	ProgmanService
http	PublicationService
http	PublishingRecordService
http	ReportingMeasureService
http	ScoringFunctionService
http	ScoringRuleService
http	SegmentService
http	SubjectService
http	TibItemService
http	UserService

IV. Test Strategy

1. Test Tools

Tools may be used to:

- Invoke business rules by manipulating, creating, deleting or updating test data
- Determine test pass/fail criteria in a manual and automated fashion
- Document requirements to traceability

2. Test Environment

The DRC test environment is located at DRC and supported by the development team. The environment uses the following software, tools and frameworks:

Tool, Framework, Software, etc.	Category	Version
Java	Programming Language	1.7
Spring Framework	Framework	3.2.0
MongoDB	Database	2.0

EC2	AWS	n/a
Oracle VirtualBox	Virtual Machine	4.2.6
Apache Tomcat	Server	7.0.35
JCE	Encryption	
Hyperic	Monitoring Agent	5.0
Servlets	Platform	2.5
Advanced Rest Client Google Chrome	Testing tool	3.1.1
Hexawise	Testing tool	n/a
cURL	Testing tool	7.2.8.1
Chrome, Firefox, IE, Safari	Browsers	
FileZilla	Testing tool – SFTP	3.6.0.2

3. Test Interface

The Test Authoring interface provides interfaced web services, search widgets, drag and drop controls and CRUD functionality. Sorting columns within Grids, assessment management, subject management, publication management, scoring functions and item selection algorithm management; In addition REST was utilized to perform stateless API tests.

V. Test Deliverables

The table below lists the deliverables providing proof of testing.

Document Name	Type
Test Plan (this document)	PDF
Test Results (this document)	PDF
Test Scripts	PDF
Test Dashboard (this document)	PDF
Test Report (this document)	PDF

Acceptance Criteria

The Test Authoring logical component interface is a cog among Assessment Creation and Management components. The user interface allows for CRUD operations for users to interact with creating, managing and publishing assessments for tenants as a user with role based permissions. Consequently, the goal of testing is not to completely demonstrate that currently documented requirements have been exactly met; rather, the goal is to demonstrate that the delivered Test Authoring component fulfills the basic purpose of the component and can be enhanced or modified as requirements are discovered and finalized. Test Authoring is the first component in the stream for publishing.

II. Introduction

2. The overall purpose of this component is to manage and maintain assessment data that is needed across the test delivery system and for the Test Spec Bank, Test Packager and other components in the system. This includes:

    a. A UI to allow for creation and management
    b. A REST service to return a web-addressable path
    c. A UI for storing system configuration data
    d. RESTful services system configuration data and integration data
    e. A Database of collections of data
    f. Authentication Integration for login and logout

Users of Test Authoring may be administrators managing content to psychometric analysis users approving packages. Clients of the Test Authoring component will be system components which interface with the services and assets; both from an upstream and downstream approach. Meaning Test Item Bank, Test Spec Bank, Test Packager, Test Administration and Test Registration.

Terms and Definitions

Term	Definition
Assessment	Data items which may include images to be displayed or included on a user interface. The data items may be associated to a component, and a tenant.
Package	The customization of the appearance of the system for a particular Consortium client. Branding assets include logos and banners.
Core Standards	The portion of Program Management that is responsible for externalizing properties and managing component endpoints.
Content	The portion of Test Authoring responsible for data management

RESTful Endpoint	The location of a service, pointed to by a URI
Program	For the purposes of these requirements, the term “program” refers to a customer’s plan for administering tests. A program is not considered to be an entity.
URI	Uniform Resource Identifier, a string of characters used for identifying and locate a resource
Tenancy	Hierarchical structured component
DTD	Document Type Definition provided by AIR
TIB	Test Item Bank
TSB	Test Specification Bank
TR	Test Registration
TAA	Test Administration

Terms and Definitions

Term	Definition
APIP	Accessible Portable Item Profile. A technical standard that focuses on accessibility in assessment items.
Blueprint	The design for a test. The test blueprint indicates the number of test questions or points related to each competency on the test and the relative emphasis placed on each competency.
Field Test	Test made up of test items intended to develop and calibrate new assessments.
Grade Level	Grade or standard the item is designed for
Item	A question on a test. It may be composed of several parts including audio and visual files.
Item Author	A person or agency responsible for item creation.
Item Bank	A systems application that manages the storage and retrieval of assessment items, tracks item versioning and lineage, and provides a robust search and query capability that allows searching on all types of metadata.
Item Status	A term used to indicate how an item can be used. Some of the valid values are: • Pilot Test • Field Test • Operational
JSON	JavaScript Object Notation, a lightweight, text based, open standard designed for human-readable data interchange.
Pilot Test	A trial series of new or modified items given to a select group of students.
Retired Item	An item that will no longer be used for an assessment.
Rubric	a guide listing specific criteria for grading or scoring academic papers, projects, or tests
Runtime exception	An unanticipated condition for which no error handling exists but normal processing cannot continue

Source Item Bank	The Item Bank from which an item is pushed to a Test Item Bank
XML	Extensible Markup Language, a set of rules for encoding documents in machine-readable form.

Data and Data Relationships

The Test Authoring user interface allows for creation, management and retrieving role based assessment information associated with deployments, components, and tenants.

	Data Utilized	Additional Information
1.	Item Data	Item data is contained in the Test Item Bank which stores items that are ready for operational or field test use on an assessment. Test Authoring performs queries on the Test Item Bank for items that meet various search criteria for allocation to an adaptive item pool or a fixed test form.
2.	Test Specification Data	Test Specification data stores various domain entities such as tests, blueprints and forms. Item data and test specifications together form the basis for test packages that can be transferred and loaded into a test delivery system for online test administration to students.
3.	Standards Publication	Test Authoring uses standards publications provided by the Core Standards component as a source for standards-related data, primarily for blueprint creation. Every test requires a standards publication

Requirements

Requirements are numbered according to the following convention:

1. RFP.## - a requirement from the RFP
2. RADPM.## - a requirement from the detailed requirements from RFP-11 or the Architecture document
3. PRPM.## - a requirement from the Proposal
4. DRPM.## - a detailed requirement from the Level I document
5. OTHPM.## - other Program Management requirements

The Test Authoring component must meet the applicable requirements from the General Requirements. The following are the DTD and example XML artifacts accompanying this document.

Req. Number	Requirement	Test Package Artifact	Description
RFP 7	Test Authoring Component will meet requirements detailed in the SBAC IT Systems Architecture (Appendix C)		Outlined in the RFP, this is the header of the main requirement for Test Packaging.
PRTIB 1	Create XML file containing data required for the DTD.	testpackage.dtd	A formal specification of the XML elements that represent a test. XML files that conform to this DTD are suitable the following purposes: 1. Administration: an XML representation of a test that contains all elements necessary to administer the test to a student. 2. An Administration XML packaged with item metadata is suitable for adaptive test simulation 3. An Administration XML packaged with full item content, metadata and item assets can be used to configure a Test Delivery system for administration of a test to students 4. Scoring: an XML representation of a test that contains sufficient elements for configuration of a test scoring system with custom rules for scale scoring this test 5.Reporting: an XML representation of a test that contains sufficient elements for configuration of a test reporting system for reporting of the scale scores for a test 6. Complete: A superset of all the test elements separately required in all of the test representations above
PRTIB 2	Create XML file containing data required for the DTD.	DE_ALG_registration.xml	An abridged test specification that is used to configure the Test Registration component. No item data is packaged in

			this format.
PRTIB 2	Create XML file containing data required for the DTD.	DE_PT_ALG_admin.xml	A test specification that is used to perform adaptive simulations or to configure a test delivery system. A simulation XML must be packaged with the item metadata, and an XML intended for a test delivery component requires full item metadata, content and all item assets such as graphics and multimedia files.
PRTIB 3	Create XML file containing data required for the DTD.	DE_read4_scoring.xml	A test specification that is used to configure a test scoring system. Only item references and scoring information is packaged in this format.
PRTIB 4	Create XML file containing data required for the DTD.	DE_read4_reporting.xml	A test specification that is used to configure a reporting system. No item data is packaged in this format.
PRTIB 5	Create XML file containing data required for the DTD.	DE_ALG_complete.xml	A test specification that is a superset of all the elements described above. The four test specification types above contain some common elements and some unique elements. The Complete package contains a superset of all elements.
PRTIB 6	A Test Packaging function must be included with the Test Authoring application to create Test Packages for various purposes. In order to create a test package explicitly using the Test Packaging user interface, the test and its sub-elements must have been fully approved and be in the Approved or Published workflow level described in section 7.2 Test Approval Workflow		

Test data at rest in the Test Authoring system are not necessarily stored in this form, but when a test package is created, the test elements must be exported as XML files that conform to this DTD.

Source.ID	Requirement	Category	Priority	Comments
RFP 6	Test Authoring Component meets requirements detailed in the SBAC IT Systems Architecture (Appendix C)	Data Format		

RADTIB 1.0	The Test authoring application will perform all validation and consistency checks to ensure that all the necessary elements are in place, and are consistent within and between each other	Validate		
RADTIB 1.1	A test must be validated before it can be submitted for approval. Successful validation transitions a test into the Validated state. A test will stay in the Not Submitted state upon unsuccessful validation	Validate		A test can also be submitted for validation at any time with no intent to move it further along in the workflow. For example, a user can perform a validation to determine if there are any missing form partitions.
RADTIB 1.2	Error and warning messages will be displayed to the user if the validation is unsuccessful that will help the user to make necessary corrections to the test specification elements.	Validate		Validation of a test is not considered to be a high privilege activity in Test Authoring. It provides a convenient means of asking the system to tell the user what remains to be done to complete the test specifications
RADTIB 1.3	A test in the Validated state can be modified at any time, so modifying the test by changing, adding or deleting any test specification element will automatically cause the test to be transitioned back to Not Submitted.	Validate		
RADTIB 2.0	Submission of a validated test for approval requires higher level user privileges than validation.	Submit		
RADTIB 2.1	Approval of a test requires higher level user privileges than submission for approval.	Submit		
RADTIB 2.2	The user that approves the test must be a different user than the one that submitted the test regardless of the level of privilege	Submit		
RADTIB 2.3	Publishing a test requires a high level of privilege as publishing affects external systems	Submit		
RADTIB 2.4	A test that has been submitted for approval and subsequently will have to be validated and resubmitted for approval by an authorized user	Submit		
RADTIB 3.0	A state in the Approved state is locked down and cannot be modified in preparation for publishing, so an	Unapprove		

	authorized user must explicitly transition the test Not Submitted via the Unapprove transition.			
RADTIB 4.0	A state in the Published state is locked down and cannot be modified, so an authorized user must explicitly transition the test Not Submitted via the Unpublish transition.	Unpublish		
RADTIB 4.1	Unpublishing a test will not change the status of any previously pushed test packages to external systems. The test will have to be transitioned through the above states and a new version must be pushed again to any external systems	Unpublish		
RADTIB 5.0	Create a version of the item selection algorithm that is provided as major.minor version string. It is acceptable to have multiple item selection algorithms with the same name so long as the versions are different. This is done to accommodate new versions of a item selection algorithm and maintain compatibility with older versions for an existing test.	Algorithm		Item selection algorithms in Test Authoring are composed of an item selection algorithm name and a variable collection of parameters. New item selection algorithms are added to Test Authoring by means of external configuration.
RADTIB 5.1	label	string		A string that describes the name of the item selection algorithm. Labels are offered to the user in a dropdown for selection.
RADTIB 5.2	version	string		v
RADTIB 5.3	number of parameters	int		Item selection algorithms have a variable number of parameters, and this field supplies the expected number of parameters
RADTIB 5.4	identifier	UUID		A UUID used to match the item selection algorithm in Test Authoring with the corresponding module in Test Delivery
RADTIB 6.0	Item selection algorithms have parameters, but this table only indicates the number of parameters Create The following parts of an item selection rule parameter.			
RADTIB 6.1	label	string		This is the name of the parameter. This will be used for labels for data entry fields in the configurable user interface

RADTIB 6.2	type	scalar | blueprint		The item selection parameter may be a scalar configuration value or may be a value that is required for each blueprint element,
RADTIB 6.3	position	int		This the sequence of presentation of the item selection rule parameter in the configurable user interface
RADTIB 6.4	type	Integer | float |boolean		The parameter can be a scalar or a dictionary. The allowable values of scalars are integer, float and string. The parameter value can also be a dictionary. Dictionaries are not of predetermined length, and allow the user to add an arbitrary number of rows. This means that the user interfaces ha to be sufficiently flexible to allow for entry of new dictionary rows which consist of an index value and a parameter value. The user interface should validate the type on data entry.
RADTIB 6.5	default	<value>		The default value that the parameter should be given when first selected by the user in the user interface
RADTIB 6.6	minimum	<value>		The minimum value the parameter should have (optional), this is validated by the user interface
RADTIB 6.7	maximim			The maximum value the parameter should have (optional), this is validated by the user interface
RADTIB 6.8	identifier	UUID		A UUID used to match the item selection algorithm parameter in Test Authoring with the corresponding parameter in Test Delivery
RADTIB 7.0	Create Scoring rules in Test Authoring that are composed of a sequential set of computation rules.			
RADTIB 7.1	name	string		A string that describes the name of the computation rule. The list of computation rule names is offered to the user in a dropdown for selection. The combination of the name and the version should be unique.
RADTIB 7.2	version	string		The version of the computation rule is provided in as major.minor version string. It is acceptable to have multiple computation rules with the same name so long as the versions are different. This is done to accommodate new versions of a computation rule and maintain compatibility with older versions for an existing test.
RADTIB 7.3	number of parameters	int		Computation rules have a variable number of parameters, and this field supplies the expected number of parameters
RADTIB 7.4	identifier	UUID		A UUID used to match the scoring rule in Test Authoring with the corresponding module in Test Delivery
RADTIB 8.0	Create computation rule parameters.			

RADTIB 8.1	name	string		This is the name of the parameter. This will be used for labels for data entry fields in the configurable user interface.
RADTIB 8.2	position	int		This the sequence of presentation of the computation rule parameter in the configurable user interface
RADTIB 8.3	parametertype	Integer | float | string | integerdictionary | floatdictionary | stringdictionary		The parameter can be a scalar or a dictionary. The allowable values of scalars are integer, float and string. The parameter value can also be a dictionary. Dictionaries are not of predetermined length, and allow the user to add an arbitrary number of rows. This means that the user interfaces ha to be sufficiently flexible to allow for entry of new dictionary rows which consist of an index value and a parameter value. The user interface should validate the type on data entry.
RADTIB 8.4	indextype	Integer, float, string		If the parameter is a dictionary, this indicates the index type. The user interface should validate the type on data entry.
RADTIB 8.5	identifier	UUID		A UUID used to match the item selection algorithm parameter in Test Authoring with the corresponding parameter in Test Delivery
RADTIB 9.0	The Test Authoring application will use the Smarter Balanced SSO component for user authentication and authorization.			This requirement documents the use of Shared Services component by Test Authoring.
RADTIB 10	The Test Authoring application will use the Smarter Balanced Permissions component to obtain a list of roles that have permissions on Test Authoring and the mapping between these roles and Test Authoring permissions.	Architecture		This requirement documents the use of Shared Services component by Test Authoring.
RADTIB 11	The Test Authoring application will use the Smarter Balanced Program Management component as a source of configuration management data to locate shared services and other components it depends on.	Architecture		This requirement documents the use of Shared Services component by Test Authoring.
RADTIB 11.1	The Test Authoring application will use the Smarter Balanced Program Management component to provide configuration management data that other components will use to interface with Test Authoring.	Architecture		This requirement documents the use of Shared Services component by Test Authoring.
RADTIB 12	The Test Authoring application will use the Smarter Balanced Monitoring and Alerting component to issue alerts and notifications.	Architecture		This requirement documents the use of Shared Services component by Test Authoring.
RADTIB 13	The Test Authoring application will use the Smarter Balanced Core Standards application as a source of standards information.	Architecture		This requirement documents the use of Shared Services component by Test Authoring.
RADTIB 14	The Test Authoring application will use the Smarter Balanced Test Item Bank	Architecture		This requirement documents the use of Shared Services component by Test Authoring.

	component as a source of item information.			
RADTIB 15	The Test Authoring application will use the Smarter Balanced Test Spec Bank component to store and retrieve Test Specification information.	Architecture		This requirement documents the use of Shared Services component by Test Authoring.
RADTIB 16	The Test Authoring application shall be multi-tenant.	User Roles and Tenancy		Tenants are configured in Program Management, and particular Test Authoring users are associated with a given tenant.
RADTIB 16.1	Test data and dependent elements shall be maintained separately for each tenant.	User Roles and Tenancy		Users will only be able to see data that is relevant to their specific tenancy and will not be able to see other tenant's test and dependent elements.
RADTIB 16.2	Tests and test elements created and managed by a given user are done so on behalf of the tenant identified by the entity associated by the user's Test Authoring role.	User Roles and Tenancy		All system roles are associated to an entity. Users logging on to Test Authoring must have a valid Test Authoring role that is associated to an entity that is a valid tenant.
RADTIB 16.3	Users that do not have a valid Test Authoring role will not be given access to the Test Authoring application.	User Roles and Tenancy		The Permissions component maps user roles to permissions that belong to a component. Only users with roles that are mapped to at least one Test Authoring role will be given access.
RADTIB 16.4	Users with a valid Test Authoring role that are not associated to a valid tenant will not be given access to the Test Authoring application.	User Roles and Tenancy		A user may have a valid Test Authoring role, but if their valid Test Authoring role is not associated with a valid tenant, then the user will not be given access.
RADTIB 16.5	Users with more than one valid Test Authoring role associated with more than one valid tenant will be given access to one and only one tenant at a time.	User Roles and Tenancy		If a user has valid test authoring roles for more than one tenant, Test Authoring must resolve the discrepancy. If it is unable to do so, the user must be presented with a choice and must select to represent one and only one tenant at a time.
RADTIB 16.6	The Test Authoring application will maintain a Setup user interface for creation and management of subject metadata and management of publication standards associations that will not be multi-tenant.	User Roles and Tenancy		Creation of subjects and subsequent association to various tenants requires that the Setup UI see across tenants.
RADTIB 17	The Authoring application shall create permissions that shall be used for fine-grained authorization.	Roles and Permissions		A permission is a unit of access that is granted to a user, and is the most granular level of access provided to a user. The granularity of permissions must be determined by the system designer.
RADTIB 17.1	Users shall be granted access to Test Authoring features ~~and~~ based on the permissions the user's role ~~is~~ has authorized ~~to~~ access to.	Roles and Permissions		All user roles are mapped to component permissions by the Permissions component, and this information is available to components through the Permissions API. Test Authoring must determine the specific user roles that are mapped to Test Authoring permissions, and when a user is authenticated, Test Authoring provides access to only those features based on the permissions that the user's role is allowed to access.

RADTIB 18	Elements originating internally to Test Authoring such as tests, segments or forms will use a unique identifier assigned by the Test Authoring application.	Identification		All elements must be uniquely identified. If the element is originated and managed by Test Authoring, it will also manage identification.
RADTIB 18.1	External elements such as content standards, items and passages that already provide unique identifiers will use the existing external unique identifier to identify the element.	Identifier		Elements that already have unique identifiers should be referred to by those identifiers.
RADTIB 19	Test specification elements at all levels will be versioned using a major.minor version scheme.	Versioning		What constitutes major and minor version changes will depend on the specific entity. Additional requirements will be added in the detailed requirements process.
RADTIB 20	The Test Authoring application will maintain Subject metadata for each tenant via a user interface that is not multi-tenant.	Subject Metadata		The list of subjects to choose from to create tests will be tenant-specific
RADTIB 20.1	Subject metadata shall include a full subject name as well as a three-character short code.	Subject Metadata		The short code is used for various purposes such as concatenation for a test name and for data exchange with other applications such as Test Delivery.
RADTIB 20.2	Both the full subject name and three-character subject code shall be emitted in the test package XML.	Subject Metadata		The short code is needed by downstream systems.
RADTIB 21	Specific standards publications from Core Standards will be associated with a tenants and a subject specific to that tenant.	Publication Standards		When selecting a standards publication for a test, a user will only be able to select standards publications that have been previously associated with the test subject for their specific tenancy. This makes it possible to choose different standards publications for the same subject for different tenants. For example, California users wanting to create a Biology test will want to use California science standards, but North Carolina users will want to choose North Carolina Biology standards.
RADTIB 22.1	Item selection algorithms shall be configurable in Test Authoring so that new item selection algorithms and their parameters can be added at any time.	Item Selection Algorithms		Test Authoring should accommodate new Item selection algorithms by means of configuration.
RADTIB 22.2	Configured versions of item selection algorithms shall have a major.minor version number.	Item Selection Algorithms		This allows us to keep old versions of item selection algorithms for compatibility while making changes in new versions
RADTIB 22.2	The Test Segment user interface shall present configured item selection algorithms for selection by the Test Author, and shall provide a dynamic user interface for entering configuration parameters specific to the selected algorithm.	Item Selection Algorithms		The user interface will respond to new item selection algorithms by presenting them as a choice and providing facilities to enter their customized parameters.
RADTIB 22.3	Test Authoring shall provide a setup function that analyzes the external inputs for configuration of item selection algorithms and provide users with appropriate error messages if something	Item Selection Algorithms		If the configuration for Item Selection algorithms is provided via an external XML file, the file will be checked for correctness and conformance to a DTD or XSD. Any error messages based on this analysis is presented to

	is wrong with the external configuration.			the user so they can make appropriate modifications.
RADTIB 22.4	The system will support a minimum of three item selection algorithms: fixedform, field test and adaptive	Item Selection Algorithms		Test Authoring will support these three item selection algorithms by default. Other item selection algorithms can be added after Test Authoring is completed by means of the extensibility mechanism described in the above requirements.
RADTIB 23	Scoring functions shall be configurable in Test Authoring so that new scoring functions and their parameters can be added at any time.	Scoring Functions		Test Authoring should accommodate new scoring functions by means of configuration.
RADTIB 23.1	Configured versions of scoring functions shall have a major.minor version number.	Scoring Functions		This allows us to keep old versions of scoring functions for compatibility while making changes in new versions
RADTIB 23.2	The Scoring Rules user interface shall present configured scoring functions for selection by the Test Author, and shall provide a dynamic user interface for entering configuration parameters specific to the selected function.	Scoring Functions		The user interface will respond to new scoring functions algorithms by presenting them as a choice and providing facilities to enter their customized parameters.
RADTIB 23.3	Test Authoring shall provide a setup function that analyzes the external inputs for configuration of scoring functions and provide users with appropriate error messages if something is wrong with the external configuration.	Scoring Functions		If the configuration for scoring functions is provided via an external XML file, the file will be checked for correctness and conformance to a DTD or XSD. Any error messages based on this analysis is presented to the user so they can make appropriate modifications.
RADTIB 24	The Test Authoring application permits authorized users to create and manage Test entities.	Tests		The Test entity is the highest level domain entity. Deleting a Test will also delete all lower level entities, so access to test elements should be carefully controlled.
RADTIB 24.1	The Test Authoring application permits authorized users to create and manage Test Segment entities.	Test Segment		This requirement documents the need for the creation and management of test segments.
RADTIB 24.2	The Test Authoring application permits authorized users to create and manage Test Blueprint entities.	Test Blueprint		This requirement documents the need for the creation and management of
RADTIB 24.3	If a non-leaf node standard is made inactive and its children are not made inactive, a message shall be displayed on each active child node: “Parent Inactive”	Test Blueprint Validation		If a standard is inactive, all of its child standards must also be inactive. However, an active standard can have inactive children.
RADTIB 25	The Test Authoring application permits authorized users to create and manage Item Pool entities.	Item Pool		This requirement documents the need for the creation and management of Item Pools.
RADTIB 26	The Test Authoring application permits authorized users to create and manage Test Form entities.	Test Form		This requirement documents the need for the creation and management of Test Forms.
RADTIB 27	The Test Authoring application permits authorized users to create and manage Performance Levels entities.	Performance Levels		This requirement documents the need for the creation and management of Performance Levels.

RADTIB 27.1	As a user I would like to record performance level information so I can include it in a test package	Performance Levels		TESTAUTHOR-424
RADTIB 28	The Test Authoring application permits authorized users to create and manage Scoring Rules entities.	Scoring Rules		This requirement documents the need for the creation and management of Scoring Rules.
RADTIB 29	The Test Authoring application permits authorized users to create and manage Reporting Measures entities.	Reporting Measures		This requirement documents the need for the creation and management of Reporting Measures.
RADTIB 30	The Test Authoring application permits authorized users to create and manage Segment Blueprint entities.	Segment Blueprint		This requirement documents the need for the creation and management of Segment Blueprints.
RADTIB 31	The Test Authoring application permits authorized users to create and manage Segment Pool entities.	Segment Pool		This requirement documents the need for the creation and management of Segment Pools.
RADTIB 32	The Test Authoring application permits authorized users to create and manage Segment Form entities.	Segment Form		This requirement documents the need for the creation and management of Segment Forms.
RADTIB 33	The Test Authoring application permits authorized users to create and manage Form Partition entities.	Form Partition		This requirement documents the need for the creation and management of Form Partitions.
RADTIB 34	The Test Authoring application permits authorized users to create and manage Item Group entities.	Item Group		This requirement documents the need for the creation and management of Item Groups.
RADTIB 34.1	items to be associated with item groups			Implicit associations i. Reference to same passage? As far as we can tell, the only thing that causes items to be grouped into an item group are they mutual association to the same passage. This means that Test Authoring should group items together into an item group if they refer to the same passage. The passage reference should be part of the item metadata. Items that have no passage references or unique passage references are automatically grouped by Test Authoring into an itemgroup.
RADTIB 35	The Test Authoring application permits authorized users to create and manage Conversion Table entities.	Conversion Table		This requirement documents the need for the creation and management of Conversion Tables.
RADTIB 30	The Test Authoring application will maintain the workflow state (Not	Workflow		Each of these levels requires independent submission and approval. The levels are

	Submitted, Validated, Submitted, Approved and Published) for the following: Registration, Administration, Scoring, and Reporting.			telection o be considered independent and not sequential.
RADTIB 30.1	The Test Authoring application will provide the means to perform the necessary validations upon user demand that will check whether a test is ready for promotion to the Registration, Administration, Scoring and Reporting states, and provides appropriate error messages and detailed information so the user can perform necessary corrections.	Workflow		This validation feature checks whether a test is ready for promotion from Not Submitted to Submitted.
RADTIB 30.2	The Test Authoring application will not permit a test to be submitted for approval that has not been validated and deemed ready for submission.	Workflow		The user must pass all validations for submitting a test for approval for one of the test purposes managed by Test authoring. A test may be deemed ready for submission for approval for Administration and Reporting and not for Administration and Scoring. This depends on the completeness and consistency of the set of required elements for promotion to Approved for that purpose.
RADTIB 30.3	The user that submits a test for approval for a given purpose cannot be the same user that approves the test for promotion from the Submitted to Approved state.	Workflow		Regardless of the role of the user, the submitter and approver must be two different users.
RADTIB 31	Deleted domain entities will be copied to a separate space for deleted entities for possible recovery	Data Retention		This is especially critical if a high-level entity is deleted from the system. This will cause cascading deletion of all dependent entities.
RADTIB 31.1	Modified domain entity data shall be copied to history tables or collections to ensure an audit history of changes are maintained and shall include the user that made the change and a time/date stamp.	Data Retention		This is necessary to track who changed test data over time.
RADTIB 32	TBD	Test Packaging		Test Packaging requirements will be added in a subsequent revision of this document.
RADTIB 33	Test Authoring requires a considerable number of consistency checks and validations to ensure that all test specification elements are complete within themselves and consistent across each other. This section captures these validations in the form of additional requirements	Test Segments		
RADTIB 34	For every test in the system, a minimum of one test segment is required	Blueprint		Test with “no” test segments are actually a single-segment test. This maintains consistency in the software and does not require any special cases. More than one test segment is certainly allowed, and the system will not impose a limit to the number of test segments because practical limits to this will typically prevail
RADTIB 34.1	The Test Blueprint screens shall implement adequate validation to ensure that the indicated mins and maxes are consistent throughout the standards	Blueprint		The values entered for blueprint screen mins and maxes must follow certain rules related

	hierarchy.			
RADTIB 34.2	The operational item min value for every content standard must be less than or equal to the operational item max value for each standard Display message: “Min must be ≤ Max”	Blueprint		If this constraint is violated, display a message.
RADTIB 34.3	The sum of the operational item min values of a higher-level content standard’s active children must be less than or equal to its operational item max value. Display message: “The max value of the content standard (N) is less than the sum if its children’s min value (M)	Blueprint		If a parent standard has a max of N items, its children cannot have a total min value M greater than N. Substitute actual values for N and M in the message.
RADTIB 34.4	The sum of the operational item max values of a higher-level content standard’s active children must be greater than or equal to its operational item min value. Display message: “The max value of the content standard (N) is smaller than the sum if its children’s min value (M)	Blueprint		If a parent standard has a min of N items, its children cannot have a total max value M less than N. Substitute actual values for M and N in the message.
RADTIB 34.5	The field test item min value for every content standard must be less than or equal to the field test item max value for each standard Display message: “Min must be ≤ Max”	Blueprint		If this constraint is violated, display a message.
RADTIB 34.6	The sum of the field test item min values of a higher-level content standard’s active children must be less than or equal to its field test item max value. Display message: “The max value of the content standard (N) is less than the sum if its children’s min value (M)	Blueprint		If a parent standard has a max of N items, its children cannot have a total min value M greater than N. Substitute actual values for N and M in the message.
RADTIB 34.7	The sum of the field test item max values of a higher-level content standard’s active children must be greater than or equal to its field test item min value. Display message: “The max value of the content standard (N) is smaller than the sum if its children’s min value (M)	Blueprint		If a parent standard has a min of N items, its children cannot have a total max value M less than N. Substitute actual values for M and N in the message.
RADTIB 35	If a standard is made inactive, a message must be displayed that indicates that the standard is inactive.	Test Forms		A visual cue is required so a user is clear that a standard has been made inactive.
RADTIB 36	A test form is required if any test segment has an item selection algorithm choice of fixedform	Form Partitions		Test forms are specifically for fixed form tests. If any test segment is marked as fixedform, at least one test form with at least one form partition is required
RADTIB 36.1	A form partition is required for every test segment with an item selection algorithm choice of fixedform			If there are N test segments marked with an item selection choice of with an item selection algorithm choice of fixedform, then N form partitions are required

RADTIB 36.2	This screen allows the creation of form partitions that belong to a form. Filters are provided to narrow the number of forms under which to create a partition.	Form Partitions		● A form requires as many form partitions as there are fixed form segments
RADTIB 36.3	Edit Form Partitions allows the user to use the search screen to search the existing item pool for items to place on the form. Items can be sequenced and deleted as well.	Edit Form Partitions		● Item searches are conducted against the item pool, not against the test item bank.
RADTIB 36.4	This screen allows the assignment of test form partitions to test segments.	Assign Form Partitions to Test Segments		● Only fixed form test segments participate in form partition assignment.
RADTIB 36.5	This screen allows the user to build scoring rules by selecting computation rules and entering the parameters of the computation rule. The computation rules are configured into the Test Authoring system and the user interface presents the configured parameters dynamically.	Scoring Rules		● The user is offered a preconfigured set of computation rules to choose from. ● Once created, computation rules can be moved up and down as the sequence of rules is critical. ● The user interface for entering the computation rule parameters is dynamic based on parameter configuration. ● Existing blueprint elements are referred to.
RADTIB 37	Roles and permissions should be managed in the following way by the Smarter Balanced Shared Services			Permissions are defined by component functions, and are an atomic unit of access control. Users that have access to a certain component permission are authorized by the component to access the feature or function. Although the Test Authoring permissions are defined by the Test Authoring system and code, the list of permissions for Test Authoring is captured by the Permissions component.
RADTIB 37.1	Roles are not defined by components and are instead managed by the Permissions component			
RADTIB 37.2	Test Authoring is responsible for determining from the Permissions component the roles that matter to Test Authoring, and how each of those roles map to established permissions in Test Authoring			
RADTIB 37.3	When a user is authenticated, Test Authoring determines which of the user's roles are applicable, and from the role, the features that this individual user is authorized to access.			

RADTIB 38	The Test Authoring permissions will grant access to screens that correspond to the test specification entities separately for CRUD operations. This means that user roles with:			Delete is combined with Create because they are considered to be equal but higher level activities than update, which is in turn higher than read. Read implies read only. The User Interface Summary matrix below will include a column for permissions for each screen.
RADTIB 37.1	Create/Delete permissions will be able to create new tests,			
RADTIB 37.2	Read permissions is able to view test data but not create, modify or delete, and			
RADTIB 37.3	Update is able to modify existing tests , but not create or delete them.			
RADTIB 38	The following table summarizes the user interface screens in the Test Authoring application.			
RADTIB 38.1	This screen provides home screen navigation access to the various Test Authoring functions	Home		Individual users will be able to access only the screens that correspond to the permissions they are authorized to access. Only the features that the user is permitted to access are available and all others are disabled.
RADTIB 38.2	This screen provides navigation access to the Setup functions	Setup		Setup is an administrative function that few users should be able to access.
RADTIB 38.3	Screens to CRUD subject metadata are required. The list of publishers offered come from here.	Subjects		● A subject cannot be deleted once a test using this publisher has been created ● A subject cannot be modified once a test using this publisher has been published
RADTIB 38.4	Standards publications from Core Standards are browsed and associated with a publisher and a subject. ● Show a grid of publications with the publication name, description and subject ● Provide a dropdown with every publisher that has a publication in the repository. Allow the user to select a publisher to filter the publications in the grid by publisher. Default is "Show all publishers" ● Show a dropdown of subjects from the subject metadata to select from ● Show a dropdown of publishers from the publisher metadata to select from	Standards Publications		● Once a test has been created with a given subject, publisher and standards publication, the publication/publisher/subject association cannot be disassociated ● Cardinality isn't a requirement, this is an item issue ● Required levels are primarily for items ● The required levels are needed only on the BP screen itself

	• Confirm association of a standards publication to a publisher and a subject. • Allow disassociation of a publication from a subject/publisher			
RADTIB 38.5	An XML configuration file for item selection algorithms is loaded on demand • The configuration file is analyzed and issues problems are reported to the user • If the configuration file has no errors, the user can accept the changes into the Test Authoring configuration	Item Selection Algorithms		• Changes cannot be accepted unless the configuration file integrity check reveals no errors • Changes to existing item selection configuration that changes an item selection algorithm already selected into a test are not allowed
RADTIB 38.6	An XML configuration file for scoring functions is loaded on demand • The configuration file is analyzed and issues problems are reported to the user • If the configuration file has no errors, the user can accept the changes into the Test Authoring configuration	Scoring Functions		• Changes cannot be accepted unless the configuration file integrity check reveals no errors • Changes to existing scoring function configuration that changes a scoring function already selected into a test are not allowed
RADTIB 38.7	• Show all tests in a grid • Provide dropdowns to filter by publisher, subject and grade • Provide a link for users to select a test to edit • Provide a means for users to delete a test • Provide the means for users to create a new test	Tests		• A test cannot be deleted once it has been published • Deleting a test destroys all child elements
RADTIB 38.8	• Show all test segments in a grid • Provide dropdowns to filter test segments by publisher, subject, grade and test • The test dropdown is itself filtered by publisher, subject and grade • Provide a link for users to select a test segment to edit • Provide a means for users to delete segment a test • Provide the means for users to create a new test	Test Segments		• A test segment cannot be deleted once it has been published • A test segment may be added, but it changes the versioning
RADTIB 38.9	• Show all test blueprints in a grid • Provide dropdowns to filter test segments by publisher, subject, grade, test and test segment • The test and test segment dropdowns are themselves filtered by publisher, subject and grade • Provide a link for users to select a test segment to edit • Provide a means for users to delete segment a test • Provide the means for users to create	Blueprints		Detailed requirements in section 9 Additional Functional Requirements and in section 10 Consistency Checks and Validations. See Test Blueprint Validations for detailed business rules for this screen.

	a new test			
RADTIB 38.10	This screen is organized by the same standards that are selected for the blueprint. ● Blueprint information is shown in a read-only manner ● Total operational and field test items in the pool ● Switch between Standards and SOCKs view ● View total number of operational and field test items ● Switch between a master test item pool or a segment item pool ● Summary of item searches are shown as item operational and field test item counts (counts roll up hierarchically) ● Item summary counts for operational and field test and for test and segment (when viewing segment pool) ● Edit links take you to an Edit Item Pools screen to the means to view items individually (all items, or items for a particular content standard and their children if any) ● Search links allow the user to search (or search again) for items for all standards or for a particular content standard and their children if any) ● Searches are conducted for both operational and field test items	Item Pools		● The Test Item Bank is searched for items that correspond to the content standards selected for the blueprint ● New searches may repopulate standards with items previously pruned from the item pool ● Searches can be conducted for any all content standards or for any individual content standard at any level in the hierarchy ● Searching for items at a content standard level higher in the hierarchy than the lowest level will include all child standards as well
RADTIB 38.11	This screen shows individual items (not just item summaries) organized by the same standards selected for the blueprint ● Toggle between viewing items and passages ● Filter items by an individual content standard (and its children) or see items for all standards ● Display of total number of items in pool, filter and segment (if editing a segment pool) ● Switch between master test pool and segment pools ● Delete individual items from the master pool ● See more details about an item	Edit Item Pools		● Cannot change OP/FT, Required or Active status of an item from the segment pool view ● Cannot delete an item from the segment pool view, only deselect it from the pool ● Cannot change passage status or delete a passage: passages are pulled in by item references
RADTIB 38.12	This screen allows the creation of test forms. Forms are simple container elements that contain form partitions.	Forms		● A test with at least one fixed form segment requires at least one form
RADTIB 38.13	This is pending as more information is required.	Conversion Tables		

RADTIB 38.14	Performance levels are a sequential set of level cutters. They refer to a blueprint element and permit the user to enter a low and high value.	Performance Levels		Existing blueprint elements are referred to.
RADTIB 38.15	This screen allows the user to create reporting measures, which are a collection of references to existing computation rules from Scoring Rules. The user adds computation rules to a reporting measure until they are satisfied.	Reporting Measures		● Existing blueprint elements are referred to ● Existing computation rules are referred to
RADTIB 38.16	The Workflow screens are left to DRC engineers to design	Workflow		
RADTIB 39	The selection of active standards on a test blueprint is a prerequisite for selection of an item pool because the item pool screen searches items according to their alignment with active standards.			The Item Pool screen has a selector to view the master test item pool, or to choose the test's child segments to view and edit the segment pool. This feature is similar to the blueprint screens. At the segment level, no items can be added to the master pool, items deleted from the segment pool are not deleted from the master item pool. However, items removed from the master item pool are also deleted from the segment item pool. Invalidation of items follows a similar pattern: it can only be done in the master item pool and not in the segment pools, and the item becomes invalid in the segment pools as well.
RADTIB 39.1	User should have the option to do the following:·			
RADTIB 39.1	Repeat more limited searches on individual standards in the pool			
RADTIB 39.2	Prune the item pool by removing items from the pool or making them inactive in the pool			
RADTIB 40	Create Item Pools (Segment Pool)			Switching from Standards to Socks in the button bar allows you to see the item pool summary of items allocated to SOCKs for the master test or to child test segments.
RADTIB 40.1	No item searches are allowed in the SOCKs view to expand the item pool. Rather, the item pool SOCK assignment can be pruned by selecting the Edit feature for a SOCK.			Screen 10-3: Item Pools (SOCKs)

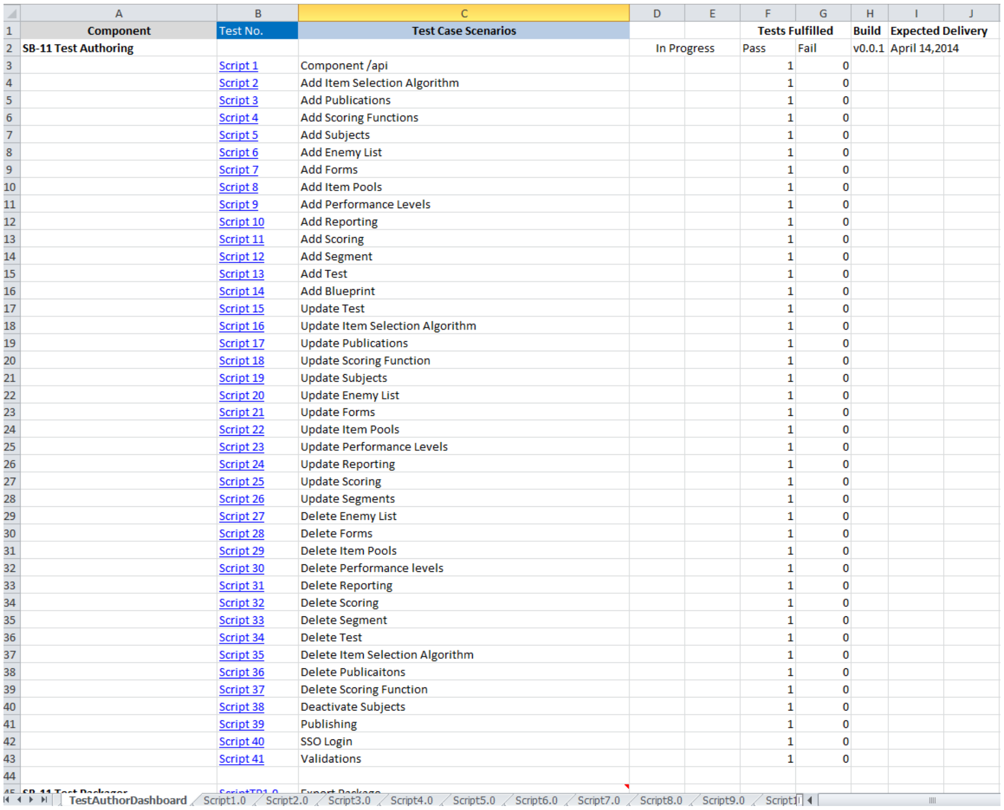
RADTIB 41	The Test Authoring application will provide a user interface so that the psychometric evaluation result records can be created and stored with the test at any time during the test workflow, including after approval or publishing.	Psychometric Information	Psychometric evaluation records do not alter any of the test's test elements.	
RADTIB 41.1	The Test Authoring application will provide a user interface that allows the attachment of arbitrary files such as text, spreadsheet or pdf files to particular psychometric evaluation result records at any time during the test workflow, including after approval or publishing.	Psychometric Information	Same as above: psychometric Evaluation records do not alter any of the test's test elements, so they can be updated at any time without changing the test.	This is straightforward and initiated in the TestAuth UI. User adds a psychometric record and provides the minimal input shown in the mockups. The upload feature permits the user to upload, attach and subsequently download any file attachment: pdf, spreadsheet, etc. This can happen even to a test that has been published and isn't locked down. - Per David Tuesday, February 25, 2014 1:58 PM
RADTIB 42.1	Test Authoring Component will meet requirements detailed in the SBAC IT Systems Architecture (Appendix C)		Outlined in the RFP, this is the header of the main requirement for Test Packaging.	
RADTIB 42.2	Create XML file containing data required for the DTD. testpackage.dtd	Test Package	A formal specification of the XML elements that represent a test. XML files that conform to this DTD are suitable the following purposes: 1. Administration: an XML representation of a test that contains all elements necessary to administer the test to a student. 2. An Administration XML packaged with item metadata is suitable for adaptive test simulation 3. An Administration XML packaged with full item content, metadata and item assets can be used to configure a Test Delivery system for administration of a test to students 4. Scoring: an XML representation of a test that contains sufficient elements for configuration of a test scoring system with custom rules for scale scoring this test 5.Reporting: an XML representation of a test that contains sufficient elements for configuration of a test reporting system for reporting of the scale scores for a test 6. Complete: A superset of all the test elements separately required in all of the test representations above	For each test specification domain entity, information about how interface elements map into XML fields emitted by the test packager. I discussed this with Jon Cohen. He told me that this mapping was a detailed design activity that DRC should do. In one of the tech calls, I discussed with Mike and the attendees. I indicated on that occasion, and subsequent occasions, that when designing and implementing any feature, the designer should: 1. Pay attention to the DTD and ensure that the UI provided all of the elements required. 2. Consult the examples and determine if there were any usage patterns of XML elements that were required but the UI didn't provide. 3. Determine what elements from the UI mapped into XML elements in the test package. In many of the tech calls, whenever a question was asked, I retraced these steps during the call with the intention of ingraining this analysis process into the development team. In many email communications, I also retraced these steps analyzing the DTD, the example XMLs and the UI mockups and indicating how they are related. Ping M
RADTIB 42.3	Create XML file containing data required for the DTD.	Test Package	An abridged test specification that is used to configure the Test Registration component. No item data is packaged in this format. DE_ALG_registration.xml	
RADTIB 42.4	Create XML file containing data required for the DTD. DE_ALG_registration.xml	Test Package	An abridged test specification that is used to configure the Test Registration component. No item data is packaged in this format.	
RADTIB 42.5	Create XML file containing data required for the DTD. DE_PT_ALG_admin.xml	Test Package	A test specification that is used to perform adaptive simulations or to configure a test delivery system. A simulation XML must be packaged with the item metadata, and an XML intended for a test delivery component requires full item metadata, content and all	

Comment [1]: Updated Test Pack information on test specification domain entity, information about how interface elements map into XML fields emitted by the test packager

			item assets such as graphics and multimedia files.	
RADTIB 42.6	Create XML file containing data required for the DTD.DE_read4_scoring.xml	Test Package	A test specification that is used to configure a test scoring system. Only item references and scoring information is packaged in this format.	
RADTIB 42.7	Create XML file containing data required for the DTD. DE_read4_reporting.xml	Test Package	A test specification that is used to configure a reporting system. No item data is packaged in this format.	
RADTIB 42.8	Create XML file containing data required for the DTD.DE_ALG_complete.xml	Test Package	A test specification that is a superset of all the elements described above. The four test specification types above contain some common elements and some unique elements. The Complete package contains a superset of all elements.	
RADTIB 42.9	A Test Packaging function must be included with the Test Authoring application to create Test Packages for various purposes. In order to create a test package explicitly using the Test Packaging user interface, the test and its sub-elements must have been fully approved and be in the Approved or Published workflow level described in section 7.2 Test Approval Workflow	Test Package		
RADTIB 43	Create CAT Simulator that uses REST interface to create a simulation record and store simulation information. The user can subsequently go into this screen and add attachments to the simulation information record.		Tests must be in the Not Submitted or Validated states to be eligible for simulation, so simulation records and attachments can only be done when the test is in these states. · Tests must be in the Not Submitted or Validated state for simulation results to be written back to the test repository	Simulation results look identical, but the simulation record isn't added by the user as with psychometric records. There are interfaces on TestAuth that the simulator needs. The sequence is as follows: 1. Simulator browses TestAuth for in-process tests ready for simulation. 2. Simulator is able to extract in-process tests for simulation. 3. Simulator performs simulations. 4. Simulator writes modified results back to TestAut bluepri so, it creates a simulation record. 5. When the tests go through the normal approval workflow, they become ineligible to be modified by the simulator. 6. Simulation records permit the upload of arbitrary attachments (PDFs, documents, spreadsheets, etc.) at any time, even after tests are published.

Comment [2]: Updated Requirements with info from David.

VI. Test Dashboard



	A	B	C	D	E	F	G	H	I	J
1	Component	Test No.	Test Case Scenarios			Tests Fulfilled		Build	Expected Delivery	
2	SB-11 Test Authoring			In Progress		Pass	Fail	v0.0.1	April 14,2014	
3		Script 1	Component /api			1	0			
4		Script 2	Add Item Selection Algorithm			1	0			
5		Script 3	Add Publications			1	0			
6		Script 4	Add Scoring Functions			1	0			
7		Script 5	Add Subjects			1	0			
8		Script 6	Add Enemy List			1	0			
9		Script 7	Add Forms			1	0			
10		Script 8	Add Item Pools			1	0			
11		Script 9	Add Performance Levels			1	0			
12		Script 10	Add Reporting			1	0			
13		Script 11	Add Scoring			1	0			
14		Script 12	Add Segment			1	0			
15		Script 13	Add Test			1	0			
16		Script 14	Add Blueprint			1	0			
17		Script 15	Update Test			1	0			
18		Script 16	Update Item Selection Algorithm			1	0			
19		Script 17	Update Publications			1	0			
20		Script 18	Update Scoring Function			1	0			
21		Script 19	Update Subjects			1	0			
22		Script 20	Update Enemy List			1	0			
23		Script 21	Update Forms			1	0			
24		Script 22	Update Item Pools			1	0			
25		Script 23	Update Performance Levels			1	0			
26		Script 24	Update Reporting			1	0			
27		Script 25	Update Scoring			1	0			
28		Script 26	Update Segments			1	0			
29		Script 27	Delete Enemy List			1	0			
30		Script 28	Delete Forms			1	0			
31		Script 29	Delete Item Pools			1	0			
32		Script 30	Delete Performance levels			1	0			
33		Script 31	Delete Reporting			1	0			
34		Script 32	Delete Scoring			1	0			
35		Script 33	Delete Segment			1	0			
36		Script 34	Delete Test			1	0			
37		Script 35	Delete Item Selection Algorithm			1	0			
38		Script 36	Delete Publicaitons			1	0			
39		Script 37	Delete Scoring Function			1	0			
40		Script 38	Deactivate Subjects			1	0			
41		Script 39	Publishing			1	0			
42		Script 40	SSO Login			1	0			
43		Script 41	Validations			1	0			
44										
45	SB-11 Test Packager	ScriptTP1.0	Export Package							

VII. Executive Test Summary

Our primary test objective was to prove Test Authoring component was capable to execute level I and level II requirements and received for required functional areas which are covered in the test plan. Component integration testing among Test Spec Bank, Test Item Bank and Test Packager in conjunction with Core Standards web services and SAML / SSO authentication across endpoints.

Component interactions, tenancy and deployment tests cover URI, REST operations, RESTful web API HTTP controllers, layers, JSON responses, database collections, UI/persistence, database and web services, authentication, roles and permissions.

Test coverage consisted of positive, negative, boundary and user scenarios based from L2 requirements for assessments, item selection algorithms, publications, scoring functions, SSO, subjects, enemy lists, forms, items, item pools, item groups, performance levels, reporting, segments, blueprints, publishing transitory hierarchical tenancy setup, inheritance of configurations, program management dependent functionality, filtering, sorting, workflow and user interface navigations.

Browser performance and compatibility UI testing was performed cross browsers using Chrome, Fire Fox, Safari and IE where default settings were tested. http: tests covered look and feel standards, services, functionality, logic and usability.

UI testing was based from supporting UI AIR guidelines and direction from their UI team experts that provided unique icons for functional areas, navigational pages and designs and layouts for Test Authoring and Test Specification Bank user interfaces.

Test Equipment / Tools – include but are not limited to: a repeatable infrastructure environment and intel core i5 laptops accessing the /api and using either Fire Fox, IE,Safari or Chrome browsers and their respective REST client plugins. Mac book pro with retina quad core i7 and a linux Ubuntu 12.04

All defect discoveries or improvements revealed during testing were quickly reported, managed and fixed in JIRA. Test coverage analysis compiles from combinatorial pairs, cobertura reports and requirements.

The heuristic test strategy, a simple model was used considering the requirements, tests, and environment and product elements.

Overall the Test Authoring component is functionally complex, integrated component that has good design, reliability, scalability, usability, and capability as a supporting features component among other shared services and for expected future component dependencies.