

UNF-ST&DARDS Documentation Release 3.0

Kaushik Banerjee, Paul Miller, and Rob Lefebvre

March 18, 2016

CONTENTS

1.1 Introduction 1.1.1 Development team 1.2 Installation and Verification 1.2.1 Bundle Contents 1.2.2 System Requirements 1.2.3 Installing UNF-ST&DARDS 1.2.4 Verifying INSTALLATION	. 1 . 2 . 2
1.1.1Development team1.2Installation and Verification1.2.1Bundle Contents1.2.2System Requirements1.2.3Installing UNF-ST&DARDS1.2.4Verifying INSTALLATION	. 2 . 2 . 2
1.2 Installation and Verification 1.2.1 Bundle Contents 1.2.2 System Requirements 1.2.3 Installing UNF-ST&DARDS 1.2.4 Verifying INSTALLATION	. 2 . 2
1.2.1 Bundle Contents 1.2.2 System Requirements 1.2.3 Installing UNF-ST&DARDS 1.2.4 Verifying INSTALLATION	. 2
1.2.2 System Requirements 1.2.3 Installing UNF-ST&DARDS 1.2.4 Verifying INSTALLATION	
1.2.2 Dystein Requirements 1.2.3 Installing UNF-ST&DARDS 1.2.4 Verifying INSTALLATION	3
1.2.4 Verifying INSTALLATION	, ບ ເ
	. 0 4
L25 Activating (Leographic Information System ((JS)	, т 13
1.2.5 Activating decigraphic information system (dis)	. 13
1.2.0 Question	. 1/
2 Architecture and Capabilities	18
2.1 System Architecture	. 18
2.2 Unified Database	. 20
2.2.1 Unified Database Content	. 20
2.2.2 Unified Database Design	. 22
2.2.3 Unified Database Software Development Kit (SDK)	. 23
2.3 Templates	. 23
2.3.1 Template Repository	. 23
2.4 Analysis Sequence	. 25
2.4.1 Fuel Depletion and Decay	. 26
2.4.2 Criticality Safety	. 26
2.4.3 Thermal Hydraulics	. 28
2.4.4 Shielding	28
2.4.5 Containment Analysis	. 28
	0
3 User Guide	29
3.1 Graphical User Interface (GUI)	. 29
3.1.1 Analysis Tab	. 29
3.1.2 Pad Maps	. 37
3.1.3 Analysis Model Geometry Viewer	. 54
3.1.4 Plot Functionality	. 57
3.1.5 SOL Viewer	. 58

		3.1.6	Reports	59
4	Deve	loper G	uide	61
	4.1	Develo	pment Environment	61
		4.1.1	Coding Standards	62
		4.1.2	Test Infrastructure	62
		4.1.3	Setting up UNF-ST&DARDS Development Environment	62
	4.2	Templa	IteEngine	65
		4.2.1	TemplateEngine Requirements	66
		4.2.2	JSON Parameter Set	66
		4.2.3	TemplateEngine Features	67
		4.2.4	Attribute and Expression Format	75
		4.2.5	Functions	75
Inc	lex			79

CHAPTER

OVERVIEW

1.1 Introduction

This technical product is a user guide of the Used Nuclear Fuel-Storage, Transportation & Disposal Analysis Resource and Data System (UNF-ST&DARDS). The objective of this manual is to help users (1) understand the various analysis processes within UNF-ST&DARDS and (2) execute various nuclear safety analyses using the graphical user interface (GUI). This manual will be updated periodically as UNF-ST&DARDS is updated. Questions or comments regarding UNF-ST&DARDS should be submitted to <UNFHelp@ornl.gov>.

UNF-ST&DARDS is being developed as a foundational resource for the DOE-NE to streamline computational analysis capabilities for characterizing the input for the overall waste management system. UNF-ST&DARDS provides a controlled, unified domestic spent nuclear fuel (SNF) system database—the Unified Database—which is integrated with nuclear analysis capabilities to support DOE waste management and fuel cycle–related objectives. The Unified Database is integrated with the analysis tools. This powerful feature of UNF-ST&DARDS provides a unique platform for performing assembly-specific and cask-specific assessments of nuclear safety based on actual characteristics of the as-loaded SNF (e.g., fuel assembly burnup, enrichment, and decay time). The cask-specific safety evaluations allow quantification of realistic safety margins associated with actual fuel loading compared with the regulatory licensing limits, and they enable a better understanding of the implications of and planning for addressing uncertainties related to aging that might arise from extended storage and subsequent transportation.

In this manual, the standard installation instructions of UNF-ST&DARDS are provided in Sect. 1 (Overview). Section 2 (Architecture and Capabilities) presents the software architecture including analysis sequences of UNF-ST&DARDS. Section 3 (User Guide) presents the GUI of UNF-ST&DARDS. Finally, Sect. 4 (Developer Guide) provides guidance to the developers including various features of the TemplateEngine.

1.1.1 Development team

The core UNF-ST&DARDS development team consists of the following ORNL scientists (listed alphabetically):

- Kaushik Banerjee <banerjeek@ornl.gov>
- Justin Clarity <clarityjb@ornl.gov>
- Rob Lefebvre <lefebvrera@ornl.gov>
- Henrik Liljenfeldt <liljenfeldth@ornl.gov>
- Paul Miller <millerpp@ornl.gov>
- Josh Peterson <petersonjl@ornl.gov>
- Georgeta Radulescu <radulescug@ornl.gov>
- Kevin Robb <robbkr@ornl.gov>
- John Scaglione <scaglionejm@ornl.gov>
- Bret van den Akker <bpvda@ornl.gov>

General questions about UNF-ST&DARDS software and methods should be directed to the email list.

1.2 Installation and Verification

The UNF-ST&DARDS installation bundled is a beta version with Unified Database access, and generated assembly type depletion libraries, and canister thermal axial heat and 3d dose maps.

Upon completion of this section you will have a functioning UNF-ST&DARDS installed.

1.2.1 Bundle Contents

This UNF-ST&DARDS installation bundle should include the following files:

UNF-STANDARDS-INSTALL-XXXXXXX.zip, and Manual.pdf (this document)

Where the XXXXXX file suffix indicates the packing date.

E.g., UNF-STANDARDS-INSTALL-2015-09-29_144036.jar indicates the bundle was created on September 29, 2015, at 144036 or 2:40:36PM.

1.2.2 System Requirements

UNF-ST&DARDS has the following minimum requirements:

- 64bit Windows 7
- Java 8 (available free at https://java.com/download)
- 4 GB Hard Disk Space
- 2 GB RAM

1.2.3 Installing UNF-ST&DARDS

If the UNF-ST&DARDS installation bundle USB is not already inserted into your USB drive, do so now.

Copy the UNF-STANDARDS-INSTALL-XXXXXXX.zip to the desired location of install. For example purposes, this section will use installation path of C:\Users\user\Desktop\UNF-STANDARDS where user is your specific user name.

Right click the UNF-STANDARDS-INSTALL-XXXXXXX.zip file and select the 'Extract all...' menu item.

Open
Extract All

After several minutes the installation will complete as indicated after which double left clicking the contained setup.bat script will produce installation specific configuration files needed for launching UNF-ST&DARDS.

] Name	Date modified	Туре	Size
🕕 data	9/29/2015 7:51 AM	File folder	
execution	9/21/2015 10:58 PM	File folder	
鷆 jre	9/29/2015 8:19 AM	File folder	
퉬 latex	9/29/2015 8:17 AM	File folder	
🌗 lib	9/29/2015 8:01 AM	File folder	
퉬 mysql	9/29/2015 8:01 AM	File folder	
🌗 UNF-Templates	9/29/2015 8:19 AM	File folder	
- Fulcrum	9/2/2015 1:44 AM	Application	34,623 KB
🗌 🚳 setup	9/28/2015 4:42 PM	Windows Batch File	2 KB
unf Type: Windows Batch File	9/28/2015 4:42 PM	RC File	2 KB
Unf Size: 1.33 KB Date modified: 9/28/2015 4:42 PM	9/28/2015 4:35 PM	Executable Jar File	8,129 KB

Once setup.bat has been executed, the launch.bat script will be available for execution. Double click the launch.bat to start UNF-ST&DARDS.

퉬 data	9/29/2015 7:51 AM	File folder	
🍌 execution	9/21/2015 10:58 PM	File folder	
鷆 jre	9/29/2015 8:19 AM	File folder	
퉬 latex	9/29/2015 8:17 AM	File folder	
퉬 lib	9/29/2015 8:01 AM	File folder	
퉬 mysql	9/29/2015 8:01 AM	File folder	
퉬 UNF-Templates	9/29/2015 8:19 AM	File folder	
🐝 Fulcrum	9/2/2015 1:44 AM	Application	34,623 KB
🐝 Fulcrum 🚳 launch	9/2/2015 1:44 AM 9/29/2015 8:57 AM	Application Windows Batch File	34,623 KB 1 KB
Fulcrum Iaunch Signature mysql_shutdown	9/2/2015 1:44 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM	Application Windows Batch File Windows Batch File	34,623 KB 1 KB 1 KB
Fulcrum Iaunch The second secon	9/2/2015 1:44 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM	Application Windows Batch File Windows Batch File Windows Batch File	34,623 KB 1 KB 1 KB 1 KB
Fulcrum Iaunch Some mysql_shutdown mysql_start Some setup	9/2/2015 1:44 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM	Application Windows Batch File Windows Batch File Windows Batch File Windows Batch File	34,623 KB 1 KB 1 KB 1 KB 2 KB
Fulcrum Iaunch Superative for the second	9/2/2015 1:44 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM 9/29/2015 8:54 AM 9/29/2015 8:57 AM	Application Windows Batch File Windows Batch File Windows Batch File Windows Batch File RC File	34,623 KB 1 KB 1 KB 1 KB 2 KB 2 KB
Fulcrum Iaunch mysql_shutdown mysql_start setup unf.rc UnfDbView	9/2/2015 1:44 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM 9/29/2015 4:35 PM	Application Windows Batch File Windows Batch File Windows Batch File Windows Batch File RC File Executable Jar File	34,623 KB 1 KB 1 KB 2 KB 2 KB 8,129 KB

1.2.4 Verifying INSTALLATION

Upon completing INSTALLING UNF-ST&DARDS you can verify your installation.

Navigate to the location to which you installed UNF-ST&DARDS.

鷆 data	9/29/2015 7:51 AM	File folder	
execution	9/21/2015 10:58 PM	File folder	
퉲 jre	9/29/2015 8:19 AM	File folder	
퉬 latex	9/29/2015 8:17 AM	File folder	
퉲 lib	9/29/2015 8:01 AM	File folder	
퉬 mysql	9/29/2015 8:01 AM	File folder	
퉬 UNF-Templates	9/29/2015 8:19 AM	File folder	
🗰 Fulcrum	9/2/2015 1:44 AM	Application	34,623 KB
Fulcrum	9/2/2015 1:44 AM 9/29/2015 8:57 AM	Application Windows Batch File	34,623 KB 1 KB
Fulcrum Iaunch mysql_shutdown	9/2/2015 1:44 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM	Application Windows Batch File Windows Batch File	34,623 KB 1 KB 1 KB
Fulcrum Fulcrum Jaunch mysql_shutdown Mysql_start	9/2/2015 1:44 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM	Application Windows Batch File Windows Batch File Windows Batch File	34,623 KB 1 KB 1 KB 1 KB
Fulcrum Iaunch mysql_shutdown mysql_start setup	9/2/2015 1:44 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM 9/29/2015 8:54 AM	Application Windows Batch File Windows Batch File Windows Batch File Windows Batch File	34,623 KB 1 KB 1 KB 1 KB 2 KB
Fulcrum Iaunch mysql_shutdown mysql_start setup unf.rc	9/2/2015 1:44 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM 9/29/2015 8:57 AM 9/29/2015 8:54 AM 9/29/2015 8:54 AM	Application Windows Batch File Windows Batch File Windows Batch File Windows Batch File RC File	34,623 KB 1 KB 1 KB 1 KB 2 KB 2 KB 2 KB

Double left-click **launch.bat** windows batch script file to launch a **cmd** console and UNF-ST&DARDS.

A **cmd** console will display the activity of UNF-ST&DARDS. Specifically, as UNF-ST&DARDS starts, it attempts to determine if the Unified Database and associated MySQL database

server are running and starts them as needed.



UNF-ST&DARDS will start on the Analysis Page.



To verify integration click the Pad Maps tab.

Menu Help	
Tables SQL Viewer Reset Selection	[site] 🔻
Sol Viewer Reset Selection Pad Maps Assembly total decay heat (by discharge date) Pad map overview Exports Assembly total decay heat (by burnup) Cask serial. Model Vendor Inservice d Registration Analysis Normal conditions of transportation - 2-meter dose Normal conditions of transportation - surface dose Cask serial. Model Vendor Inservice d Registration Cask criticality Cask total decay heat Cask clad temperatures Series name Use a "not" filter Use a "Not" filter Show All Show None Show None Show None	License

Select the Maine Yankee site from the **[site]** drop.

Menu Help Tables SOL Viewer Pad Maps Assembly total decay heat (by discharge date) Exports Analysis Normal conditions of transportation - personnel boundary dose Normal conditions of transportation - 2-meter dose Cask surface temperatures H + b I	🛃 UNF ST&DAF	NDS								×
Tables Reset Selection Isite SQL Viewer Reset Selection Isite Pad Maps Assembly total decay heat (by discharge date) Isite Exports Assembly total decay heat (by discharge date) C + e Manaysis Normal conditions of transportation - personnel boundary dose C + e Normal conditions of transportation - zmeter dose Normal conditions of transportation - zmeter dose C + e Normal conditions of transportation - zmeter dose C + e E + e Normal conditions of transportation - zmeter dose C + e E + e Cask surface temperatures Hypothetical accident condition - 1-meter dose E + e Cask criticality Cask total decay heat Cask clat temperatures H + e Series name Use a "Not" filter McGuire 1 McGuire 2 Milistone 1 Milistone 2 Milistone 3 Monticelio Milistone 3 Show None Monticelio Morris	Menu Help									
Pad Maps Assembly total decay heat (by discharge date) Exports Assembly total decay heat (by burnup) Normal conditions of transportation - personnel boundary dose Cask serial. Normal conditions of transportation - 2-meter dose D + Cask surface temperatures Pad map overview Cask surface temperatures Cask total decay heat Series name I + Use a "not" filter McGuire 1 Milistone 1 Milistone 2 Milistone 3 Show None	Tables SQL Viewer						Reset Selection	[sit	e] "	1
	SQL Viewer Pad Maps Exports Analysis	Assembly total decay heat (by discharge date) Assembly total decay heat (by burnup) Normal conditions of transportation - personnel boundary dose Normal conditions of transportation - surface dose Normal conditions of transportation - surface dose Normal conditions of transportation - surface dose Cask surface temperatures Hypothetical accident condition - 1-meter dose Cask criticality Cask total decay heat Cask criticality Cask total decay heat Series name	Pad map overv Cask serial]	Nodel	Vendor	Inservice	Action Registration	[sit	e] *	
							Ŧ			

Upon selection, the data tabs will be populated for viewing.

🔬 UNF ST&DAF	RDS											
Menu Help												
Tables												
SQL Viewer							Res	et Selectio	n Maine Yank	ee 🔻 [cask] 🔻	
Dad Mans						-						
Exporte	Assembly total deca	y heat (by discharg	e date)	2	<u>}</u>	Pad map overview		_				
Applie		Assembly total de	cay heat	(by burnup)		Cask serial number	Model	Vendor	Inservice date	Regis	tration	
Analysis	Normal cor	ditions of transport	ation - pe	ersonnel boundary dose		TSC-24-TSC-1	TSC-24	NAC	Nov 8, 2002	ML06332	20236 🔺	
	Norm	al conditions of trai	nsportati	on - 2-meter dose		TSC-24-TSC-10	TSC-24	NAC	Mar 7, 2003	ML06332	20236 =	
	UNF ST&DARDS Menu Help Tables SQL Viewer Pad Maps Exports Analysis Normal conditions of transportation - pe Normal conditions of transportation Normal conditions of transportation Cask surface temperatures Hypothetical ac Cask criticality Cask total decay heat (Disc 24-TSC-10 W TSC-24-TSC-10 W TSC-24-TSC-11 W TSC-24-TSC-12 W			on - surface dose		TSC-24-TSC-11	TSC-24	NAC	Feb 8, 2004	ML06332	20236	
	Normal conditions of transportation - 2-inter dose Normal conditions of transportation - 2-inter dose Cask surface temperatures Hypothetical accident condition - 1-m Cask criticality Cask total decay heat Cask clad temp Series name Cask criticality				080	TSC-24-TSC-12	TSC-24	NAC	Dec 2, 2002	ML06332	20236	
	Cask sufface temps	Cask total door	w hoat	Cask alad tomporatur	036	150-24-150-13	TSC-24	NAC	Dec 29, 2002	ML06222	20230	
	Cask chucality	Cask total deca	ynear	Cask clau temperature	85	TSC-24-TSC-14	TSC-24	NAC	Dec 14, 2002	ML 06332	20236	
	Series name			Cask Criticality		TSC-24-TSC-16	TSC-24	NAC	Feb 21 2003	ML 06332	20236	
	TSC-24-TSC-1	▲ US		eerad		TSC-24-TSC-17	TSC-24	NAC	Mar 24, 2003	ML 06332	20236	
	✓ TSC-24-TSC-10	= i	1 1		-	TSC-24-TSC-18	TSC-24	NAC	Oct 26, 2002	ML06332	20236	
	TSC-24-TSC-11	L	• •			1						
	TSC-24-TSC-12	· · · · · · · · · · · · · · · · · · ·			-			10				
	✓ TSC-24-TSC-13	- 19										
	✓ TSC-24-TSC-14	× 15		,			L L	Jse a "not"	filter			
	Tables L Viewer Assembly total decay heat (by discharge date) Assembly total decay heat (by discharge date) Normal conditions of transportation - personnel boundary dose Normal conditions of transportation - sensonnel boundary dose Normal conditions of transportation - sensonnel boundary dose Normal conditions of transportation - sensonnel boundary dose Cask surface temperatures Hypothetical accident condition - 1-meter dose Cask curface temperatures Hypothetical accident condition - 1-meter dose Cask curface temperatures Hypothetical accident condition - 1-meter dose Cask curface temperatures Cask clad temperatures Show None V TSC-24-TSC-10 V T											
	Service Reset Selection Maine Yankee [cask] Assembly total decay heat (by discharge date) Image: Cask serial number Pad map overview Cask serial number Cask serial number Cask serial number Image: Cask seri											
	Use a not filter		+ +			Maine Yankee Wet Storage:						
	Show All											
	Chaw Mana	113	-		-	Maine Yankee ISFSI:						
	Show None					NAC-UMS UMS-24:						
	Assembly total decay heat (by discharge date) Assembly total decay heat (by burnup) Normal conditions of transportation - personnel boundary dose Normal conditions of transportation - 2-meter dose Normal conditions of transportation - 1-meter Cask surface temperatures Hypothetical accident condition - 1-meter Cask criticality V TSC-24-TSC-10 V TSC-24-TSC-10 V TSC-24-TSC-11 V TSC-24-TSC-12 V TSC-24-TSC-12 V TSC-24-TSC-12 V TSC-24-TSC-14 V TSC-				Vendor: NAC Interna	tional Inc.						
		-		Date		Loaded Canisters: 64	areana, anc.					
		 MAP NO. MAP NO. MAP NO. MAP NO. MAP NO. MAP NO. 	11 + 1614 1612 + 16 28 + 1614 1612 + 16	9-162 • 169-162 • 169-169 • 169-165 • 169-162 • 16 9-162 • 169-162 • 169-169 • 169-165 • 169-162 • 16	24 W.2	Loaded Assemblies: 1	424				-	
		 MANDA + MANDA + MANDA 	EI + 3024-303 + 30	1972 1 1 109 102 1 109 103 1 109 109 1 109 105 1 10	awa a	Dealth Assemblies: 1	434					
		• X2/K4 • X2/KC • X2/	14 • 323 324 • 3	19 201 + 20	2470.04	Position: Vertical						
		• X2+X6 • X2+X6 • X2+7	10 + 104309 - 1	194969 - 1824964 - 1824969 - 1824967 - 1824969 - 182	180	Coc: 72-1015					-	

Select the cask labeled TSC-24-TSC-1 from the [cask] drop down.

🔬 UNF ST&DAR	RDS				
Menu Help					
Tables SQL Viewer			Reset Selection	Maine Yankee 🔻	[cask] 🔻
Exports Analysis	Hypothetical accident condition Normal conditions of transportation Surface temperatures Clad temperatures Axial thermal map Canister models Cask criticality Total decay heat	Cask details Field Serial number		TSC-24-TSC 1	
		Model Cask in-service date Licensed decay heat load k-eff at in-service date Peak clad temp at time of loading Cask max surface temp at time of Cask min surface temp at time of Cask loaded weight	floading	TSC-24-TSC-70 TSC-24-TSC-11 TSC-24-TSC-12 TSC-24-TSC-13 TSC-24-TSC-14 TSC-24-TSC-15 TSC-24-TSC-16	
	View criticality model View shielding model	Assembly position and overview Position Contents Initial E	Enrichment Disch	TSC-24-TSC-17 ar TSC-24-TSC-18	arge Dat

Cask-specific data will be loaded and available for viewing.

🔬 UNF ST&DARE	DS										
Menu Help											
Tables								ſ	1.0		
SOI Viewer								Reset Sele	ection Maine Yan	kee TSC-24-TS	-1 * [assembly]
Dad Mane							_				
Exporte	Axial thermal map	Hypothetical a	accident	condition Norma	al conditions of transporta	ion	Cask detai	ls			
Exports	Canister models	Cask critical	lity	Total decay heat	Surface temperatures	Clad temperatures		Fie	eld		Value
Analysis	ĺ						Serial num	nber		TSC-24-TSC-1	
							Model			TSC-24	
							Cask In-se	ervice date		Nov 8, 2002	
							Licensed	becay neat loa	30	0.66217	
							Peak clad	temp at time	of loading	353 592	
							Cask max	surface temp	at time of loading	91.167	
							Cask min s	surface temp	at time of loading	89.379	
							Cask loade	ed weight			
							Assembly	position and	overview		
							Position	Contents	Initial Enrichment	Discharge Burgun	Discharge Date
							1	B003	2.41	12276.0	Jun 29, 1974
							2	A065	2.04	11069.0	Jun 29, 1974
							3	EF0085	2.52	30391.0	Jan 11, 1980
							4	N858	3.3	37328.0	Apr 7, 1990
							5	J863	3.0	36802.0	Mar 31, 1984
							6	EF006L	2.51	30360.0	Jan 11, 1980
							7	C305	2.95	14884.0	May 2, 1975
							8	K042	3.0	35150.0	AUG 17, 1985
			1	Many addition life a	and at		10	P020 M857	3.5	44137.0	Oct 15, 1988
				view criticality in	nodel		11	M416	3.3	36781.0	Oct 15, 1988
							12	B050	2.4	16135.0	May 2, 1975
				View shielding h	nodel		13	EF005R	1.94	18367.0	Apr 9, 1977
							14	Q442	3.69	34250.0	Feb 14, 1992
							15	LT71	3.29	41988.0	Mar 28, 1987
							16	N416	3.3	42063.0	Apr 7, 1990
							17	K015	3.0	37796.0	Aug 17, 1985
							18	C234	2.90	30200.0	May 2, 1975
							20	H208	3.03	36126.0	Sep 24, 1982
							21	M435	3.3	38956.0	Oct 15, 1988
							22	EF006V	2.52	30200.0	Jan 11, 1980
			23	C215	2.94	16525.0	May 2, 1975				
							24	B017	2.4	12159.0	Jun 29, 1974
							View can	ister reference			
							Site-Spec	ific Euel - R	enresented as sta	andard design basis	s fuel assembly
							configural Modeled a thermal	tion with spe as a repres	ent fuel assembly entative assembly	initial enrichment a with spent fuel ass	nd burnup for critic embly decay heat f
							_				

Click the **View shielding model** button to generate the cask-specific shielding model and render it in the **Fulcrum** graphical user interface. This can take a few seconds as the Unified Database is queried for all parameters needed for construction of the shielding model.

Tables Reset Selection Table Selection <th>g oral procerna</th> <th>:DS</th> <th></th>	g oral procerna	:DS											
Tables Rest Selection Maine Yankee* TsC.24.TSC.1* [assembly] Value remotes Cask criticality Total decay heat Surface temperatures Clad temperatures Field Value Value Analysis Cask criticality Total decay heat Surface temperatures Clad temperatures Field Value Value Analysis Cask criticality Total decay heat Surface temperatures Clad temperatures Field Value Value Analysis Cask criticality Total decay heat Surface temperatures Clad temperatures Clad temperatures Clad temperatures Total decay heat Value Value Total and the ender decay heat Value Value Value Value	lenu Help												
QL View Reset Selection Maine Yanke * TSC.24.TSC.1 * [assembly] Avial thermal map Analysis Hypothetical accident conditions of transportation Cask criticality Cask criticality Total decay heat Surface temperatures Claid temperatures Claid temperatures Claid temperatures Field Value Analysis Cask criticality Total decay heat Surface temperatures Claid temperatures Claid temperatures Claid temperatures Claid temperatures Field Value Value Analysis Cask criticality Total decay heat Surface temperatures Claid temperatu	Tables								[
Pad Mapa Axial thermal map Hypothetical accident conditions of transportation Canister models Cask critically Total decay heat Surface temperatures Clad temperatures Serial number TiSC-24-TISC-1 Model TISC-24-TISC-1 Model TISC-24-TISC-1 Model TISC-24-TISC-1 Model TISC-24-TISC-1 Model TISC-24-TISC-1 Model TISC-24-TISC-1 Model TISC-24-TISC-1 Model TISC-24-TISC-1 Model TISC-24-TISC-1 Least In-service date No.8, 2002 Licensed decay heat load Licensed decay heat load Left TI-service date TISC-24-TISC-1 Model TISC-24-TISC-1 Cask In-service date TISC-24-TISC-1 Model TISC-24-TISC-1	SQL Viewer								Reset Sel	ection Maine Yan	kee * TSC-24-TSC	-1 * [assem	ibly]
Deports Anatyris Avaid therms may Anatyris Hypothetical accident condition Normal conditions of transportation Canister models Cask criticality Total decay heat Surface temperatures Clad temperatures Surface temperatu	Pad Maps	-	K				7						_
Canister models Cask criticality Total decay heat Surface temperatures Cled temperatures Field Value Berlai number TSC-24-TSC	Exports	Axial thermal map	Hypothetical accide	nt condition	Normal	conditions of transportation		Cask detail	5				_
View critically model 18:2:2:18:0:1 18:2:2:18:0:1 View critically model 18:2:2:18:0:1 18:2:2:18:0:1 View critically model 18:2:2:18:0:1 18:2:2:18:0:1 View shielding model 18:2:2:18:0:1 18:2:2:18:0:1 View shielding model 18:2:2:18:0:1 18:0:2:18:0:1 View shielding model 18:0:2:18:0:1 18:0:2:18:0:1 View shielding model 18:0:2:18:0:1 18:0:2:18:0:1 View shielding model 18:0:2:18:0:1 18:0:2:18:0:1 10:3:19:0:13:18:0:13	Analysis	Canister models	Cask criticality	Total decay	heat	Surface temperatures	Clad temperatures		Fie	eld	700.04 700.4	Value	
View criticality model View shielding model View shielding model View shielding model View shielding model View shielding model 24 1132 1133 367810 0415, 1982 View shielding model View shielding model 24 1975, 1982 113 1671, 1980 View shielding model View shielding model 25 333, 3672, 1982 3633, 1982 View shielding model View shielding model 1 1003, 24, 1 12776, 0 Jun 29, 1974 3 Erooga 24, 1 12276, 0 Jun 29, 1974 3 24, 20, 20, 20, 20, 20, 20, 20, 20, 20, 20	ritury or o							Serial num	ber		TSC-24-TSC-1		
View criticality model View criticality model<								Caskin-se	nice date		Nov 8, 2002		
View criticality model View criticality model View criticality model View criticality model View criticality model View criticality model View criticality model View criticality model View criticality model View criticality model View criticality model View criticality model View criticality model View shielding model View criticality model View criticality model View criticality model View shielding model View shielding model View shielding model View shielding model View shielding model View shielding model View shielding model View shielding model View shielding model View shielding model View shielding model View shielding model View shielding model View shielding model View shielding model View shielding model View shielding model View shielding model View shielding model								Licensed d	lecav heat loa	ad	11010, 2002		
View criticality model View shielding model 35.5.82 View shielding model 93.79 Cask mas surface temp at time of loading 99.379 Cask inade weight Cask ind overview Assembly position and overview 18003 2 Alobe 2 244 112276.0 Jun 29.1974 3 EF0025 252 303910 Jan 11.1980 4 N858 3.3 37328.0 Apr.7 1990 5 Ja863 3.0 36802.0 Mar 31.1984 6 EF0025 255 14884.0 Mar 2.1975 8 K042 3.0 36802.0 Mar 31.1984 6 EF0026 251 14884.0 Mar 2.1975 7 8 K042 3.0 36800.0 Jan 11.1980 7 23.0 C151.1988 11 M416 3.3 36781.0 Oct 15.1988 10 M875 3.4 3.0 3600.0 Jan 17.1980 13 14 0442.30.0 For 14.1992.1977 13 EF005R 13.4 1030.7.0 Apr 9.1977 13 EF0058 </th <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <th></th> <td>k-eff at in-s</td> <td>ervice date</td> <td></td> <td>0.66317</td> <td></td> <td></td>								k-eff at in-s	ervice date		0.66317		
View criticality model View criticality model<								Peak clad t	emp at time	of loading	353.582		
Cask min surface temp at time of loading P3.79 Cask to loaded weight Assembly position and overview Assembly position and overview Position Contents Initial Enrichment Discharge Burnup Discharge Date 1 B003 2.41 112276.0 Jul 29, 1974 2 A005 2.04 110690 2.04 110690 Jan 11, 1980 4 N858 3.3 37328.0 Apr 7, 1990 5 Ja63 3.0 36602.0 Mar 31, 1984 6 EF0005 2.52 30391.0 Jan 11, 1980 7 C.305 2.95 14884.0 Mar 2, 1975 8 K042 3.0 35151.0 Aug 7, 1992 10 Mar 3.3 35781.0 Oct 15, 1988 11 M416 3.3 35771.0 Oct 15, 1988 11 M416 3.3 35781.0 Oct 15, 1988 12 B050 2.4 11635.0 Mar 2, 1977 13 EF005R 1.94 116367.0 Apr 9, 1977 14 Oct 2.5 20 30200.0 Jan 11, 1980 16 N416 3.3 35781.0 Mar 2, 1975 17 K015 3.0 37780.0 Mar 2, 1975 18 C224 2.96 1177 1.84 1188 2 EF005W 2.52 302000 Jan 11, 1980 2 H208 3.0 4 H208 3.0 2 H208 3.0 2 H208 3.0 2 H208 3.0 2 H208								Cask max	surface temp	at time of loading	91.167		
View criticality model View shielding model View shielding model View shielding model								Cask min s	surface temp	at time of loading	89.379		
View criticality model View criticality model View criticality model View criticality model View shielding model								Caskloade	ed weight				_
View criticality model View criticality model Contents Initial Enrichment Discharge Date View criticality model 1 24065 2.64 11090 Jun 29, 1974 2 A065 2.64 11090 Jun 29, 1974 3 EF0885 2.62 203918 Jan 11, 1680 4 N855 3.0 36602.0 Mar 31, 1984 6 EF090L 2.51 33680.0 Jan 71, 1990 5 J863 3.0 36602.0 Mar 31, 1984 6 EF090L 2.51 33680.0 Jan 71, 1990 7 C305 2.95 14884.0 May 2, 1975 8 KO42 3.0 35161.0 Aug 2, 1975 9 P028 3.5 44137.0 Fe14, 1992 10 M857 3.3 39781.0 Od 15, 1988 11 M416 3.3 36781.0 May 2, 1975 13 EF005R 194 1935.0 May 2, 1975 14 <								Assembly	position and	overview			
View criticality model 1 B003 2.41 12276.0 Jun 29. 1974 2 2065 2.04 10909 Jun 29. 1974 3 2 2065 2.04 10909 Jun 29. 1974 3 3 EF0855 2.52 203916 Jun 11. 1980 4 Na858 3.3 3728.0 Apr.7. 1990 5 JJ663 3.0 36602.0 Mar.31. 1984 6 EF006L 2.51 30300.0 Jan 11, 1980 7 C'30.5 2.95 1.4884.0 Mag.2, 1975 8 K/042 3.0 35150.0 Aug.7, 1990 9 P/028 3.5 44137.0 F614.1992 10 M476 3.3 339730.0 Oct 15, 1988 12 B050 2.4 16135.0 Mag.2, 1975 13 EF005R 1.4 1337.0 Apr.1990 14 O442 3.99 34290.0 F614.1992 15 L171								Position	Contents	Initial Enrichment	Discharge Burnup	Discharge Da	ate
View criticality model 2 A065 2.04 11099.0 Jun 29, 1974 View criticality model 3.3 37328.0 Apr.7, 1990. 4 N858 3.3 37328.0 Apr.7, 1990. View criticality model 5 Jø63 3.0 36804.0 May 7, 1975. View criticality model 6 EF0084. 2.51 36380.0 Jan 11, 1980. View shielding model 7 2.35 44137.0 Fe15, 1988. 11 View shielding model 10 M657 3.3 4592.30 Oci 15, 1988. 12 B050.0 2.44 16135.0 May 2, 1975. 13 EF0058. 1.94 16367.0 Apr 9, 1977. 14 O442 3.0 35741.0 Oci 15, 1988. 12 B050.0 Feb 14, 1992. 197.5 15 L1771 3.29 41988.0 May 2, 1975. 14 C442.3.69 3.0 Apr 9, 1977. 16 Nati 16 3.3 37786.0 Aug 2, 1975. 15 16								1	B003	2.41	12276.0	Jun 29, 1974	
View criticality model View criticality model View criticality model View criticality model View shielding model View criticality model V				2	A065	2.04	11069.0	Jun 29, 1974					
View criticality model 4 N858 3.3 37282.0 Apr 7, 1990 5 J863 3.0 36802.0 Mar 31, 1984 6 EF006L 2.51 33680.0 Jan 11, 1980 7 C305 2.95 14884.0 Mar 2, 1975 8 K042 3.0 35750.0 Aug 17, 1985 9 P028 3.5 44137.0 Pot 14, 1992 10 M657 3.3 43922.0 Oct 15, 1988 12 8050 2.4 16135.0 Mar 2, 1975 13 EF005R 1.94 19357.0 Apr 9, 1977 14 Q442 3.0 37780.0 Apr 9, 1977 15 L717 3.29 41988.0 Mar 28, 1987 16 N416 3.3 37950.0 Apr 9, 1977 16 N416 3.3 37950.0 Apr 9, 1977 16 N416 3.3 37950.0 Apr 9, 1977 16 N416 3.3 37950.0				3	EF0085	2.52	30391.0	Jan 11, 1980					
View criticality model View shielding model View sh				4	N858	3.3	37328.0	Apr 7, 1990	-				
View criticality model 7 C305 2.95 14884.0 May 2, 1975 B K042 3.0 35150.0 Aug 17, 1985 B K042 3.0 351510.0 Aug 17, 1985 I0 M857 3.3 43923.0 Oct 15, 1988 View shielding model 10 M857 3.3 43923.0 Oct 15, 1988 12 8050 2.4 16135.0 May 2, 1975 13 5670367 1494 14370 Oct 15, 1988 14 O442 3.69 34250.0 Feb 14, 1992 15 14171 3.0 37786.0 Aug 17, 1985 16 N416 3.3 42053.0 Apr 7, 1990 17 K015 3.0 37786.0 Aug 17, 1985 18 C234 2.96 13778.0 Aug 17, 1985 186 12 198 1908 190 20 Apr 7, 1980 192 198 192, 1925 198 192, 1925 198 192, 192, 192 198 192, 192, 192, 192, 192, 192 <td></td> <td></td> <th></th> <td>5</td> <td>5803</td> <td>2.61</td> <td>30302.0</td> <td>Mar 31, 1984</td> <td></td>					5	5803	2.61	30302.0	Mar 31, 1984				
View criticality model 8 Ko42 3.0 95150.0 Aug 37, 1985 View shielding model 10 M557 3.3 43922.0 Oct 15, 1988 View shielding model 11 M416 3.3 36781.0 Oct 15, 1988 12 2055.0 2.4 116135.0 May 2, 1975 14137.0 Aug 37, 1985 13 EF005R 1.94 18367.0 Aug 47, 1997 14 0.423.99 34250.0 Feb 14, 1992 14 0.42 3.0 37978.0 Aug 17, 1985 16 Aug 37, 1985 15 L.711 3.29 41988.0 Mar 28, 1987 16 N416 3.3 32050.0 Aug 17, 1985 16 C234 2.96 13778.0 Aug 17, 1985 18 C234 2.96 Mar 28, 1987 19 EF0069 2.52 30200.0 Jan 11, 1980 20 H435 3.3 39656.0 Oct 15, 1988 20 H435 3.3 39656.0 Oct 15, 1988 23								7	C305	2.95	14884.0	May 2, 1975	
View criticality model 9 PO28 3.5 44137.0 Feb 14, 1992 Wew criticality model M857 3.3 3923.0 Oct 15, 1988 View shielding model 11 M416 3.3 39781.0 Oct 15, 1988 12 B050 2.4 16135.0 May 2, 1975 13 EF009R 1.94 13367.0 Apr 9, 1977 14 O442 3.99 34250.0 Feb 14, 1992 15 L171 3.29 41988.0 Mar 28, 1987 16 N416 3.3 42063.0 Apr 9, 1977 16 N416 3.4 2063.0 Apr 7, 1990 17 K015 3.0 37996.0 Aug 17, 1985 16 C234 2.96 13778.0 May 2, 1975 18 EF0088 2.52 30200.0 Jan 11, 1880 20 1942.0 Sep 24, 1082.2 21 M435 3.3 39656.0 Oct 15, 1988 22 14035 3.3 39656.0 Oct 15, 1988 23 C215 2.								8	K042	3.0	35150.0	Aug 17, 1985	
View criticality model View shielding shieldi								9	P028	3.5	44137.0	Feb 14, 1992	
View shielding model View shie				View crit	icality mo	del		10	M857	3.3	43923.0	Oct 15, 1988	
View shielding model 12 B050 2.4 16135.0 May 2, 1975 13 EF005R 1.94 113367.0 Apr 8, 1977 14 O442 3.69 34250.0 Feb 14, 1992 15 L171 3.29 41988.0 Mar 28, 1987 16 N416 3.3 42063.0 Apr 7, 1990 17 N615 3.0 37796.0 May 2, 1975 18 C234 2.96 13778.0 May 2, 1975 19 EF0089 2.52 30200.0 Jan 11, 1980 20 H208 3.03 35125.0 Sep 24, 1982 21 M435 3.3 39856.0 Oct 15, 1988 22 EF0089 2.52 30200.0 Jan 11, 1980 23 C215 2.94 16525.0 May 2, 1975 24 D017 2.4 1525.0 May 2, 1975								11	M416	3.3	36781.0	Oct 15, 1988	-
13 EP005H 1.94 18367.0 APD 9, 197 14 Q442 3.69 3.4250.0 Fe14, 1992 15 L1771 3.29 41968.0 Apr 9, 197 15 L1771 3.29 41968.0 Apr 7, 1990 16 N416 3.3 42063.0 Apr 7, 1990 17 K015 3.0 37796.0 Aug 7, 1993 18 C234 2.96 13778.0 May 2, 1975 19 EF0069 2.62 30200.0 Jan 11, 1980 20 H208 3.3 39956.0 Oct 15, 1988 21 M435 3.3 39950.0 Oct 15, 1988 22 EF0069 2.82 30200.0 Jan 11, 1980 23 C215 2.94 16525.0 May 2, 1975 24 E017 2.4 15159.0 May 2, 1975				View shi	elding mo	del		12	B050	2.4	16135.0	May 2, 1975	-
14 2442 3.08 342,000 Point, 1892 15 L171 3.29 41988.0 Mar 2, 1997 16 N416 3.3 42063.0 Apr 7, 1990 17 K015 3.0 37796.0 May 2, 1975 18 C234 2.96 13778.0 May 2, 1975 19 EF0089 2.52 30200.0 Jan 11, 1800 20 H208 3.03 38128.0 Sep 24, 1982 21 M435 3.3 38956.0 Oct 15, 1988 22 EF0089 2.52 30200.0 Jan 11, 1800 23 C215 2.94 16525.0 May 2, 1975 24 B017 2.4 1525.0 May 2, 1975								13	EF005R	2.60	18307.0	Apr 9, 1977	-
16 N416 2.3 4205.0 Apr.7, 1990 17 K015 3.0 37796.0 Aug.17, 1985 18 C234 2.96 13778.0 Mag.2, 1975 19 EF0089 2.52 30200.0 Jan 11, 1980 20 H208 3.0 35125.0 Sep 24, 1982 21 M435 3.3 38956.0 Oct 15, 1988 22 EF0089 2.52 30200.0 Jan 11, 1980 23 C215 2.94 15525.0 May 2, 1975 24 6017 2.4 15525.0 May 9, 1974			R					15	1.171	3.09	41988.0	Mar 28, 1987	-
17 K015 3.0 37796.0 Aug 17, 1985 18 C234 2.96 13778.0 May 2, 1975 19 EFD089 2.52 30200.0 Jan 11, 1980 20 H208 3.03 39426.0 Sep 24, 1982 21 M435 3.3 39956.0 Oct 15, 1988 22 EFD089 2.52 32000.0 Jan 11, 1980 21 M435 3.3 39956.0 Oct 15, 1988 22 EFD089 2.52 32000.0 Jan 11, 1980 23 C215 2.94 16525.0 May 2, 1975 24 B017 2.4 1525.0 Jan 9, 1974								16	N416	3.3	42063.0	Apr 7, 1990	
18 C234 2.96 1377.80 May 2.1975 19 EF0089 2.52 302000 Jan 11.180 20 H208 3.03 36126.0 Sep 24, 1982 21 M435 3.3 38956.0 Oct 15, 1988 22 EF0089 2.52 30200.0 Jan 11.180 23 C215 2.94 16525.0 May 2.1975 74 B017 2.4 12550.0 Jan 9.1974								17	K015	3.0	37796.0	Aug 17, 1985	
19 EF0089 2.52 30200.0 Jan 11, 1980 20 H208 3.03 39125.0 Sep 24, 1982 21 M435 3.3 39956.0 Oct 15, 1988 22 EF006V 2.52 30200.0 Jan 11, 1980 23 C215 2.94 15525.0 May 2, 1975 24 P017 2.4 12559.0 Jan 9, 1974							18	C234	2.96	13778.0	May 2, 1975		
20 H208 3.03 38128.0 Sep 24, 1982 21 M435 3.3 39856.0 Oct 15, 1988 22 EF000V 2.52 30200.0 Jan 11, 1980 23 C215 2.94 16525.0 May 2, 1975 24 P017 2.4 12159.0 Jan 14, 1980								19	EF0089	2.52	30200.0	Jan 11, 1980	
I21 µA435 1,5,3 38956.0 Oct 15, 1998 I22 EF008V 2.52 30200.0 Jant 11, 1980 I23 C215 2.94 16525.0 May 2, 1975 I24 PD17 2.4 12159.0 Jap 1974				20	H208	3.03	35125.0	Sep 24, 1982					
23 C215 2.94 16525.0 May 2, 1975 24 R017 2.4 12250.0 May 2, 1975					21	FE006V	3.3	30300.0	Jap 11 1090				
24 B017 2 4 12159 0 Lin 291974				23	C215	2.94	16525.0	May 2, 1975					
					24	B017	2.4	12159.0	Jun 29, 1974				
										View can	ister reference		
View canister reference								Site-Spec configurat Modeled a thermal.	ific Fuel - R ion with sp as a repres	tepresented as sta ent fuel assembly i entative assembly	ndard design basis nitial enrichment a with spent fuel ass	fuel assembly nd burnup for c embly decay he	ritic eat f
View canister reference Site-Specific Fuel - Represented as standard design basis fuel assembly configuration with spent fuel assembly initial enrichment and burnup for critic Modeled as a representative assembly with spent fuel assembly decay heat thermai.			Genera	tina shieldina r	model for	TSC-24-TSC-1							

The progress bar will indicate **Generating shielding model for TSC-24-TSC-1...** with the **cmd** console windows providing detailed activity listing the shielding model construction. Additionally, because **Fulcrum** is an external application that communicates with UNF-ST&DARDS the potential exists that Fulcrum is not running. A ConnectException, as seen in the **cmd** console, will be produced when UNF-ST&DARDS starts **Fulcrum**.



Upon successfully constructing the shielding model, it is displayed in Fulcrum.



We can now render the shielding model using Fulcrum's built-in geometry viewer by left clicking the **Visualize** button.

TSC-3	24-TSC	-1_boun	ding_NC1	(DoseRa	te_11-08	-2002.inp 🔀								
Сору	Cut	Paste	Undo	Redo	Find	document	▼ F	un E	Expand aliases	Input listing	Mixing table	Calculate geometry volumes	Visualize	
1	'Th	e re	fere	nce	date	for the	mble	so	urce dec	ay time	ə is 01-	-01-2001	Visualize	^
2	'th	imbl	e so	urce	deca	y time	(yrs) i:	s 1.8507	78713210	013005			

The default Top (X-Y) 2D cross-section through the middle of the cask will be rendered.

SCALE		And south of Annual 1982	
File View Run Help			
lavigation 🗗 🗙	TSC-24-TSC-1_bounding_NCTDoseRate_11-08-2002.inp	TSC-24-TSC-1_bounding_NCTDoseRate_11-08-2002.inp : geometry (Line 2195) 🔯	
Filter	Top (X-Y) Front (X-Z) Side (Y-Z) 3D Meshes		
Filter	Top (XY) Front (X-Z) Side (Y-Z) 30 Meshes Material Zoom in (Zx) Zoom out (Zx) 		
	X: -321.5490811; Y: 102.2331892		Origin: (0, 0, -1.9435),

We want to verify the Dose Rate Map Pack was properly installed, so we will load a Dose Rate Map for the selected Cask by clicking the **Mesh** menu button and right clicking and selecting **Load mesh file**.

TSC-24-TSC-1_bounding_NCTDoseRate_11-08-2002.inp	TSC-24-TSC-1_bounding_NCTDoseRate_11-08-2002.inp : geometry (Line 2195)
Top (X-Y) Front (X-Z) Side (Y-Z) 3D Meshes	
Available meshes Min bound Max bound Custom r	nin bound Custom max bound
Load mesh file	
Remove mesh file	
Reset	
Material Toom in (2x) Zoom out (2x)	

You will be prompted with a file browser where you need to navigate to the C:\Users\user\Desktop\UNF-STANDARDS\data\dose directory, where user has been replaced with your user name.

k in:	📙 Z: \data \dose			- 3 0	0	1 🙂 (
🐚 My	Name	Size	Туре	Date Modified		
k raq	FO-DSC-FO24P-P01_bounding_NCTDoseRate_03-01-2025.total.3dmap	1.7 MB	3dle	9/15/2:08 PM		
	FO-DSC-FO24P-P02_bounding_NCTDoseRate_03-01-2020.total.3dmap	1.7 MB	3dle	9/15/2:20 PM		
	FO-DSC-FO24P-P02_bounding_NCTDoseRate_03-01-2025.total.3dmap	1.7 MB	3dle	9/15/2:08 PM		
	TSC-24-TSC-1_bounding_HACDoseRate_03-01-2024.total.3dmap	834 KB	3dle	8/15/2:36 PM		
	TSC-24-TSC-1_bounding_HACDoseRate_03-01-2029.total.3dmap	834 KB	3dle	8/15/2:50 PM		
	TSC-24-TSC-1_bounding_NCTDoseRate_03-01-2024.total.3dmap	1.5 MB	3dle	8/9/20:28 PM		
	TSC-24-TSC-1_bounding_NCTDoseRate_03-01-2029.total.3dmap	1.5 MB	3dle	8/9/20:34 PM		
	TSC-24-TSC-2_bounding_HACDoseRate_03-01-2024.total.3dmap	834 KB	3dle	8/12/2:18 PM		
	TSC-24-TSC-2_bounding_HACDoseRate_03-01-2029.total.3dmap	834 KB	3dle	8/12/2:54 PM		
	TSC-24-TSC-2_bounding_NCTDoseRate_03-01-2024.total.3dmap	1.5 MB	3dle	8/9/20:12 PM		
	TSC-24-TSC-2_bounding_NCTDoseRate_03-01-2029.total.3dmap	1.5 MB	3dle	8/9/20:36 PM		
	TSC-24-TSC-3_bounding_HACDoseRate_03-01-2024.total.3dmap	834 KB	3dle	8/15/2:30 PM		
	TSC-24-TSC-3_bounding_HACDoseRate_03-01-2029.total.3dmap	834 KB	3dle	8/15/2:56 PM		
	TSC-24-TSC-3_bounding_NCTDoseRate_03-01-2024.total.3dmap	1.5 MB	3dle	8/9/20:28 PM		
	TSC-24-TSC-3_bounding_NCTDoseRate_03-01-2029.total.3dmap	1.5 MB	3dle	8/9/20:40 PM		
name:	TSC-24-TSC-1 bounding NCTDoseRate 03-01-2024 total 3dman					Open

Select the **TSC-24-TSC-1_bounding_NCTDoseRate_03-01-2024.total.3dmap** file and left-click the **Open** button. **HAC** and **NCT** stand for Hypothetical Accident Conditions and Normal Conditions of Transportation respectively.

The Fulcrum mesh manager will now indicate the associated 3dmap Dose Rate file with the shielding model document loaded. Expand the Responses and select **Response 1**.

TSC-24-TSC-1_bounding_NCTDoseRate_11-08-2002.inp	TSC-1_bounding_NCTDose	Rate_11-08-2002.inp : geo	metry (Line 2195) 🔀	
Top (X-Y) Front (X-Z) Side (Y-Z) 3D Meshes				
Available meshes	Min bound	Max bound	Custom min bound	Custom max bound
 TSC-24-TSC-1_bounding_NCTDoseRate_03-01-2024.total.3dmap Responses 				
Response 1	6.738471580885141e-03	1.349166532751194e+08		
Response 1 - abs. uncert.	4.194149576237037e-04	1.231954906373474e+08		
Response 1 - rel. uncert.	6.582687628704737e-03	9.240327011410117e-01		
Material Zoom in (2x) Zoom out (2x)				

The dose rate will be loaded into Fulcrum and can now be visualized as an overlay on the geometry by updating the **Material** selection to be **Overlay**.

Material 🔹
Material Material + outline Outline
Overlay Overlay + boundaries

The overlay view presents the 3d dose data overlayed on the geometry.



These steps can be reproduced for casks for which Dose Rate results are available.

You have now verified UNF-ST&DARDS has successfully been installed by launching UNF-ST&DARDS, requesting visualization of a specific cask requiring UNF-ST&DARDS to query the Unified Database, construct a cask-specific shielding input by expanding appropriate templates. And lastly, viewing installed Dose Rate results.

1.2.5 Activating Geographic Information System (GIS)

UNF-ST&DARDS out of the box has the GIS pad map view inactive due to potential graphics card hardware support issues. Activating the GIS map view is easy and can be done by updating the UNF-ST&DARDS resource file. If you followed the instructions INSTALLING UNF-ST&DARDS the resource file will be located at C:\Users\user\Desktop\UNF-STANDARDS\.unf.rc, where user is your user name.

With UNF-ST&DARDS not running, edit the file with your Windows WordPad application.

B > C = Docume ■ ■ Home View	ent - WordPad	
Clipboard	I I I A A IF I	
Open	puter ▶	
Organize	Hard Disk Drives (2) OSDisk (C:) 43.8 GB free of 297 GB → DATA_DRIVE_1 (D;) 494 GB free of 931 GB	Î
C Libraries Documents J Music Pictures S Pictures	Devices with Removable Storage (1) DVD RW Drive (E:) Network Location (1)	E
Homeoroup Fi	UNF-STANDARDS-CD (\RAQW7\Users\raq\Desktop)(W:) ile name: CAUSers\raq\Desktop\UNF-STANDARDS\.unf.rc I All Wordpad Documents (*. Open Can	v Intf,∵v ncel

In order to activate the GIS pad map view, change the **unf.map.display** property from **false** to **true**.

Your **unf.map.display** should now be **true**.



Save the file, and restart UNF-ST&DARDS. If your graphics card hardware supports the necessary components in UNF-ST&DARDS, the GIS map view will appear.



Consult the UNF-ST&DARDS Help documentation (press **F1** key or click Help>View Help) for assistance with interacting with the GIS map view.

If your graphics card hardware does not support the necessary components used by UNF-ST&DARDS, UNF-ST&DARDS will quit upon selection of the Pad Maps tab and you will see the following message in your **cmd** console.

C:\Windows\system32\cmd.exe	×
<pre>at javax.swing.JComponentpaintImmediately(Uhknoum Source) at javax.swing.RepaintManager\$3.run(Uhknoum Source) at javax.swing.RepaintManager\$3.run(Uhknoum Source) at javax.swing.RepaintManager\$3.run(Uhknoum Source) at javax.swing.RepaintManager\$3.run(Uhknoum Source) at javax.swing.RepaintManager\$3.run(Uhknoum Source) at javax.swing.RepaintManager.paintDirtyRegions(Uhknoum Source) at javax.swing.RepaintManager.paintDirtyRegions(Uhknoum Source) at javax.swing.RepaintManager.paintDirtyRegions(Uhknoum Source) at javax.swing.RepaintManager.prefaintDirtyRegions(Uhknoum Source) at javax.swing.RepaintManager.prefaintDirtyRegions(Uhknoum Source) at javax.swing.RepaintManager.prefaintDirtyRegions(Uhknoum Source) at javax.swing.RepaintManager.access\$1100(Uhknoum Source) at javax.swing.RepaintManager.access\$1100(Uhknoum Source) at java.awt.event.InvocationEvent.dispatchKUnknoum Source) at java.awt.EventQueue.dispatchKUnknoum Source) at java.awt.EventQueue.sispatchEventImpl(Uhknoum Source) at java.awt.EventQueue.sispatchEventImpl(Uhknoum Source) at java.awt.EventQueues\$3.run(Uhknoum Source) at java.awt.EventQueues\$3.run(Uhknoum Source) at java.awt.EventQueues\$3.run(Uhknoum Source) at java.security.AccessController.doPrivilege(Uhknoum Source) at java.awt.EventQueues\$3.run(Uhknoum Source) at java.awt.EventDispatchThread.pumpEventsForFilterz(Uhknoum Source) at java.awt.EventDispatchThread.pumpEventsForFilterz(Uhknoum Source) at java.awt.EventDispatchThread.pumpEvents(Uhknoum Source) at java.awt.EventDispatchThread.pumpEvents(Uhknoum Source) at java.awt.EventDispatchThre</pre>	
Unable to Start Program. Computer does not meet minimum graphics requirements. Please install up-to-date graphics driver and try again. Reason: glActiveTexture not available This program will end when you press OK.	ш
C:\Users\rag\Desktop\UNF-STANDARDS\execution>_	-

Reverting your C:\Users\user\Desktop\UNF-STANDARDS\.unf.rc file to have unf.map.dispaly=false will restore default pad map behavior.

1.2.6 Question

UNF-ST&DARDS is deployed with help documentation to assist users. The help documentation is available via the **Help>View Help** menu item. Additionally, context specific help is available by pressing the **F1** key.

Further assistance can be acquired by emailing UNFHelp@ornl.gov

ARCHITECTURE AND CAPABILITIES

2.1 System Architecture

UNF-ST&DARDS is illustrated in Fig. 2.1 and consists of six primary elements: (1) a user interface (graphical as well command line), (2) the Unified Database, (3) a Unified Database software development kit, (4) a template repository that contains constructs for specific nuclear safety analysis tools that have been integrated, (5) an application-agnostic template engine, and (6) a process manager that handles interactions between the different elements. The nuclear safety analysis tools and their user interfaces are externally developed to function independently, but they have been integrated into UNF-ST&DARDS for automating different analysis streams. The current nuclear safety analysis tools consist of SCALE¹, a comprehensive modeling and simulation suite for nuclear safety analysis and design, and Coolant Boiling in Rod Arrays-Spent Fuel Storage (COBRA-SFS)², a thermal-hydraulic analysis code. SCALE's Fulcrum user interface provides input editing and validation, data visualization, and output viewing, and it has been integrated into UNF-ST&DARDS to provide SCALE input preview, as well as model and results visualization. The software infrastructure has been designed with the capability to integrate other tools, tool interfaces, and analysis packages in a similar fashion, but this would require development of appropriate templates for inclusion in the template repository, as well as identification of specific input attributes to ensure they are present in the Unified Database or added as necessary. The UNF-ST&DARDS process manager was developed to manage (1) the workflow across the Unified Database, (2) integrated nuclear safety analysis tool templates and applications, and (3) results retrieval and interrogation. This software uses the Java Persistence application programming interface (API) (JPA) entity, controller, and annotation classes to abstract the communication with the Unified Database. The SNF templates are implemented, in conjunction with structured parameter sets and TemplateEngine, to abstract the management of nuclear safety and systems processes. UNF-ST&DARDS uses the Java programming language to provide cross-platform capabilities

¹ SCALE: A Comprehensive Modeling and Simulation Suite for Nuclear Safety Analysis and Design, ORNL/TM-2005/39, Version 6.1, Oak Ridge National Laboratory, Oak Ridge, Tennessee, June 2011.

² T.E. Michener, D. R. Rector, J. M. Cuta et al., COBRA-SFS: A Thermal-Hydraulic Code for Spent Fuel Storage and Transportation Casks, PNL-10782, Pacific Northwest National Laboratory, Richland, Washington, 1995.



and increase developer throughput.

Fig. 2.1: UNF-ST&DARDS architecture.

UNF-ST&DARDS is a three-tier application consisting of a presentation, process, and data tier. In a three-tier application, the presentation tier is the top tier, providing display of available services and results, typically in a GUI. The process or business tier is the middle tier, providing access to process functionality and controls. The data tier is the bottom tier, providing data access and persistence, and it often includes an API or a software development kit (SDK). The UNF-ST&DARDS data tier is composed of the Unified Database and its SDK. The process tier coordinates the nuclear safety applications and the Unified Database. The presentation tier is a GUI that allows point-and-click access to analysis capabilities via the process tier, as well as data extraction and visualization.

The process tier includes a command line interface for access to execute nuclear safety applications with a concise set of input parameters. The input parameters are described in JavaScript Object Notation (JSON) ³. JSON is a standard, human readable, lightweight (i.e., very few format constructs) data interchange format. The command line interface provides access to launch all UNF-ST&DARDS analysis capabilities. In the GUI, an analysis page allows the user to select an analysis type (e.g., criticality) and either (1) reactor type batch (e.g., all pressurized water reactors [PWRs]), or (2) specific analysis. The GUI is described Sect. 5.

³ ECMA-International, The JSON Data Interchange Format, ECMA-404, CH-1204 Geneva, October 2013.

2.2 Unified Database

The Unified Database provides a comprehensive, controlled source of technical data for various waste management system analysis/evaluation tools, as well as fuel cycle system analyses and safeguards and security studies. The Unified Database has been designed to be flexible and expandable to provide controlled inputs to a variety of tools and applications. Interface controls, relevant data, and data formats will continue to be identified and added as additional requirements for other tools and capabilities are identified.

Development of the Unified Database involves a staged database approach that includes the following database stages:

- Development contains databases used for development purposes, such as structural change and new data set import verification and validation
- Production the database from which input parameters for all processes are acquired and their results imported
- Test the database used for continuous integration (CI), containing the minimum set of data required to test UNF-ST&DARDS
- Deployed databases that have been released (i.e., versioned snapshots of the production database)

With these stages, the Unified Database development team can work in clearly delimited areas, eliminating the potential of affecting production capabilities while conducting development tasks.

2.2.1 Unified Database Content

The following datasets are stored in the Unified Database:

Facility attributes include utility, site, reactor, pool, and independent spent fuel storage installation (ISFSI) data. **Federal government radioactive waste attributes** include government-owned high-level radioactive waste (HLW) and government-owned SNF data. **Future facility attributes** include interim storage facility (ISF), repackaging facility, and repository data. **Transportation infrastructure attributes** include rail, heavy-haul truck, legal-weight truck, and barge assets data, in addition to transfer times between these transportation modes. **Assembly-specific attributes** include actual and projected assembly discharge information and characteristics (e.g., isotopic concentrations, source terms). **Cask/canister attributes** include overpack, capacity, physical properties, unit processing times, unit costs, and characteristics (e.g., dose, component temperatures, keff). **Economic attributes** include the unit cost of reactor sites, transportation infrastructures, ISF, repackaging facility, and repository surface costs.

These data have been collected from the DOE Office of Civilian Radioactive Waste Man-

agement, the General Council (GC) nuclear data forms (RW-859/GC-859)¹ database, and other sources. The data not obtained from the RW-859 database were obtained by having subject matter experts data mine various sources to extract the required information. These sources include DOE studies, open literature, vendor data, and utility data. One essential requirement of the Unified Database is that all of the data within the database must be traceable. All data references are uploaded to the Centralized Used Fuel Resource for Information Exchange (CURIE) website at curie.ornl.gov, allowing permanent storage and simple retrieval for all imported references. Correlating the data from the subject matter experts to the appropriate reference source is initially handled with the use of Microsoft Excel spreadsheets that include the data and web links to the references stored on CURIE prior to import into the database. The subject matter experts are responsible for verifying that the data and the reference sources concur.

A data importer was developed to upload the Excel spreadsheets to the Unified Database. The data importer uses the JPA entity, annotation, and controller classes to generate the appropriate modifications to the Unified Database and to successfully conduct data imports. The importer uses an extension convention to identify the type of data being imported. For each data record created, a reference record is also created that contains the source of the import, such as the Excel spreadsheets populated by the subject matter experts.

Data that have been imported to the Unified Database continue to be editable. Any changes to the data require review and approval before they are implemented into the production version of the database. Once changes are made to the production version, the data that have been changed and the rationale for the changes are logged and distributed with the next deployment of the database.

The Unified Database change control process records the subject matter expert's name and archives the previous data within the database. Storing the data, notes, references, and archived data is accomplished by grouping the tables into three main areas: "data," including data tables with data necessary for analysis tools; "reference," including tables of reference information corresponding to specific data tables, and "history," including tables that serve as change logs which record users and modification times of specific data and reference tables, along with the data and references that were changed.

The reference tables reflect the data tables, with the exception that each data column is a text field containing the reference as a uniform resource identifier (URI). In addition to reflecting each table's data column with a reference URI, each reference column has a corresponding modification date column indicating the time at which the data and their associated references were imported into the database. A history table exists for each reference and data table with the log of table changes. Specifically, history tables reflect the associated tables column for column, along with the time of the change, the type of change (e.g., "update," "delete"), and the identification of the user making the change.

¹ "RW-859 Nuclear Fuel Data," US Department of Energy, Energy Information Administration, Washington, D.C., October 2004.

2.2.2 Unified Database Design

The Unified Database design was created using the MySQL Community edition database. MySQL Community edition is a freely available, well-documented, open source, relational database server that uses SQL. Many utilities exist to assist in development and maintenance of MySQL databases, making it a productive environment to streamline the data needs of nuclear safety and systems analysis tools.

SQL databases are composed of tables, each of which contains a fixed number of named columns. Each table row constitutes a "record," many of which can exist in a given table. "Attributes" exist that describe each column in a table. These attributes are often the type of data in the column (integer, real, text, date, etc.), as well as the constraints (unique, auto-incrementing, optional, table relations, etc.). The SQL database, with its structured and relational constructs, lends itself to data normalization. The process of data normalization involves separating tables into their constituents until the table represents one and only one component. Normalization removes redundant data and minimizes data dependencies.

The Unified Database employs normalization as a general best practice in data management. Due to normalization, each record in a table is uniquely identifiable, creating what is called a primary key. Relationships between tables are described with a foreign key field that stores the primary key of the related table.

With hundreds of tables and thousands of table columns, nomenclature consistency is of paramount importance. The Unified Database uses a consistent convention for naming tables and columns to ensure consistent use and interpretation of the data and their relationships. Columns use a verbose name with a suffix indicating the unit of measure. For example, the assembly table contains assembly-specific attributes, one of which is the initial uranium mass. These attributes use a verbose name and the column data type (e.g., kilograms), as in *initial_uranium_kg*. Relationship columns include the related table's name and the primary key column name separated by an underscore. The assembly decay results table has a relationship with the assembly listed in the assembly table (assembly) from which the results were derived; the column that stores the foreign key is named *assembly_id*, from which the table (assembly) and primary key (*id*) of the related table are easily discernable.

In addition to the consistent nomenclature and the suffix for the unit of measure, the Unified Database includes a data dictionary to provide a more detailed explanation of what each table and column contains. The data dictionary stores a description for each column and a full name for the unit of measure. As an example, the entry for column *reactor_id* on the cycle table explains, "Foreign key to the reactor table indicating the reactor to which this cycle belongs." For a user viewing an unfamiliar table, this information can help clarify any questions about what the data mean.

2.2.3 Unified Database Software Development Kit (SDK)

One of the key features of the Unified Database is the capability for integration with multiple computational tools and models. Integration with the Unified Database is streamlined with a Java programming language, SDK. The majority of the Unified Database's SDK consists of automatically generated JPA classes. JPA interfaces the concrete data storage medium (MySQL) from the abstract data relationships and includes the Java Persistence Query Language (JPQL). The JPA classes consist of entities, annotations, and controllers. The entity classes represent the data tables. The annotation classes are used by JPQL to provide programmatic (i.e., via software), storage agnostic, conditional entity retrieval. The controller classes provide a means to create, edit, and delete entities. An advantage of the JPA is that significant portions of user-experienced SQL runtime errors are migrated to developer-mitigated compile-time errors. Compile-time errors are easier to manage by developers and help to ensure that the system is more stable for users.

2.3 Templates

UNF-ST&DARDS uses templates that serve as building blocks from which applicationspecific input is defined. The UNF-Templates repository provides UNF-ST&DARDS application-agnostic input generation. Templates allow UNF-ST&DARDS to use the same set of parameters to succinctly communicate to each application while decoupling UNF-ST&DARDS from any specific application. The TemplateEngine, which processes and expands templates, is written in the Java programming language and includes an SDK and standalone Java executable. The SDK facilitates the TemplateEngine integration into UNF-ST&DARDS, while the standalone Java executable provides a means by which template creators can test and verify their templates. All the functionalities of the TemplateEngine are described in Appendix A. The parameter set is defined as a structured collection of attributes, objects, and arrays of attributes. Objects and arrays can be embedded within one another and themselves. JSON is the syntax that represents a parameter set.

2.3.1 Template Repository

The UNF-Templates repository uses the Mercurial Distributed Version Control System (DVCS) to track changes, manage collaboration between template developers, and mitigate conflicts between developer's changes. Mercurial is integrated with FogCreek's Fogbugz management system, which features issue tracking, discussion forums, customer relationship management, and evidence-based scheduling to further enhance project management and collaboration. Current tools with integrated templates are the SCALE code system and COBRA-SFS. The template repository structure is as follows:

• input templates for SCALE's CSAS6 criticality safety analysis sequence,

- common templates used by different analysis types (e.g., header templates, file copy, etc.),
- input templates for containment analysis,
- input templates for SCALE's Transport Rigor Implemented with Time-dependent Operation for Neutronic (TRITON) fuel assembly depletion tool,
- input templates for SCALE's ORNL Isotope Generation and Depletion Code (ORI-GEN) Assembly Isotopics (ORIGAMI) analysis sequence for computing fuel assembly discharged and decayed isotopic inventory and decay heat,
- the regression test directory containing JSON files for testing,
- input templates for SCALE's MAVRIC shielding and dose analysis sequence,
- input templates for COBRA-SFS thermal analysis, and
- COBRA Creator Of Most Assembly's Necessary Data, Even Radgen (COMANDER) utility template for generating fuel assembly geometry subtemplates for COBRA-SFS thermal analysis.

These templates use Kitware's CMake, the cross-platform, open-source build and test system, in conjunction with the SCALE regression harness, to construct a UNF-Templates test harness that helps prevent regression in automated template execution capabilities. UNF-Templates testing needs the TemplateEngine and SCALE regression harness to be available. The TemplateEngine is needed for the expansion of the templates and subsequent verification of template intent. SCALE is needed for its regression harness, which drives the TemplateEngine, dissects the result, and compares it against the expected result.

A template expansion test case is created and placed into the regression test directory (UNF-Templates/regression/input). CMake is executed to create a platform-specific configuration of the test in the testing directory. Once configured, the test is verified, generating a .table file containing extracted points of interest which are subsequently compared against the expected results via a specialized difference tool provided by the SCALE regression harness. Once the result from the test case is as expected, the resulting .table file is moved into the expected results directory (UNF-Templates/regression/output). Upon future regression testing, the testing harness has an approved test result to compare against to demonstrate that the acceptance criteria have been satisfied, resulting in a pass or signaling a regression. The regression tests examine the functional requirements of the system. All new templates and their tests are added to the UNF-Templates Mercurial repository and pushed back to the central development repository for integration into UNF-ST&DARDS and or other templates.

The Wiki page (inside ORNL network) presents (1) the template repository cloning process from the central repository to a local machine for template development and (2) processes for pushing and pulling changes from the central repository. Access to the template repository requires permission from the system administrators and access to the Oak Ridge National Laboratory (ORNL) internal network. The permission process can be initiated by sending an email to UNFHelp@ornl.gov.

The criticality analysis–specific template development process is described in Ref. ¹. Similarly, the thermal ², shielding ³, and containment ⁴ template development processes are documented.

2.4 Analysis Sequence

UNF-ST&DARDS currently provides the following nuclear safety analyses capabilities:

- Depletion and decay: provides SNF isotopic compositions (subsequently used in criticality analysis), heat load (subsequently used in thermal analysis), radiation source (subsequently used in dose/shielding calculations), and isotopic activities in curies (subsequently used in containment calculations),
- Criticality: calculates the keff for as-loaded casks,
- Thermal: provides cladding and surface temperatures of as-loaded casks,
- Shielding: provides radiation dose rate maps outside the as-loaded transportation casks, and
- Containment: calculates allowable leakage rate for as-loaded transportation casks.

The current integrated nuclear safety analysis tools that perform all of the above functions (excluding containment) are SCALE and COBRA–SFS. SCALE provides the ORIGEN ORIGAMI sequence, the Criticality Safety Analysis Sequence with KENO-VI (CSAS6), and the Monaco with Automated Variance Reduction using Importance Calculations (MAVRIC) shielding sequence, which are used for performing the depletion and decay, criticality, and shielding analyses, respectively. The COBRA-SFS thermal hydraulics application is used with the COBRA-COMANDER utility ¹ to provide system component temperatures, including peak and minimum clad temperatures, as well as cask surface temperatures. The containment analysis equations are implemented in templates, and the TemplateEngine expands these templates with cask-specific parameters to determine cask-specific allowable leakage rate.

UNF-ST&DARDS process manager contains the business logic (i.e., rules on how data can be created, displayed, stored, and changed) to manage the preparation, execution, and

¹ COBRA-COMANDER cycle 1.0, Input Generator for COBRA-SFS, ORNL/TM-2014/9 Oak Ridge National Laboratory, Oak Ridge, Tennessee, March 2014.

¹ J.B. Clarity et al., Criticality Process, Modeling and Status for UNF-ST&DARDS, FCRD-NFST-2015-000440, Oak Ridge, TN, July 2015.

² K.R. Robb et al., Report Documenting Model Development for Thermal Analyses, ORNL/LTR-2013/434, Oak Ridge, TN, September 2013.

³ G.Radulescu et al., Models for Dose Rate Analyses, FCRD-NFST-2014-000358, Oak Ridge, TN, August 2014.

⁴ G.Radulescu et al., Containment Analysis Capability of UNF-ST&DARDS, FCRD-NFST-2015-000657, Oak Ridge, TN, March 2015.

results retrieval of these applications and their subsequent processes. Process management and results are formally described in the unified modeling language (UML) sequence diagram depicted in Fig. 2.2 In general, an application's process flow is as follows:

- 1. Retrieve the entities required for process execution from the Unified Database.
- 2. Retrieve the process input template.
- 3. Create a process input by expanding the process template with a JSON parameter set describing the entities.
- 4. Execute the process with the expanded process input.
- 5. Upon completion of processing, extract and store the results.

2.4.1 Fuel Depletion and Decay

The fuel assembly depletion and decay process requires assembly characteristics to execute. The process manager aggregates these characteristics (e.g., assembly-specific initial enrichment and uranium mass, irradiation history information, and discharge burnup) from the Unified Database, creating a parameter set. The fuel assembly depletion and decay template is acquired from the UNF-Templates repository and provided to the TemplateEngine with the parameter set, creating a problem- and application-specific input file. The application-specific input file to initiate the fuel assembly depletion and decay process is then executed by the process manager. This process results in generation of assemblyspecific isotopic concentrations, decay heat, and radiation source term results that are extracted and stored for future use. The application's entire result set is archived, and specific parameters or subsets are imported into the Unified Database.

2.4.2 Criticality Safety

The cask criticality safety process requires cask type, cask inventory position, cask content characteristics, and an evaluation date to execute. The process manager aggregates these characteristics from the Unified Database and creates a parameter set. The fuel assembly inventory depletion and decay prerequisite processes are executed for each assembly in a given cask if the results are not available at the evaluation date. The criticality template is acquired from the UNF-Templates repository and provided to the TemplateEngine with the parameter set, creating a problem- and application-specific input file. The application-specific input file required to initiate the cask criticality analysis process is then executed. This process generates cask-specific neutron multiplication factor (k-effective) and uncertainty results that are extracted and stored for future use.





2.4.3 Thermal Hydraulics

The thermal hydraulics analysis process requires cask type, cask inventory position, cask content characteristics, and an evaluation date to execute. The process manager aggregates these characteristics from the Unified Database, creating a parameter set. Cask-specific fuel assembly inventory depletion and decay prerequisite processes are executed if results are not already available for the given evaluation date. The thermal hydraulics template is acquired from the UNF-Templates repository and provided to the TemplateEngine with the parameter set, creating a problem- and application-specific input file. The application-specific input file to initiate the cask thermal hydraulics analysis process is executed by the process manager. This process results in generation of assembly-specific peak and minimum clad temperatures, cask-specific surface temperatures, and total decay heat results that are extracted and stored for future use.

2.4.4 Shielding

The cask shielding process requires cask type, cask position, cask inventory characteristics, and an evaluation date to execute. The process manager aggregates these characteristics from the Unified Database, creating a parameter set. Cask-specific fuel assembly and component inventory depletion, decay, and activation prerequisite processes are executed if results are not already available for the given evaluation date. The shielding template is acquired from the UNF-Templates repository and provided to the TemplateEngine with the parameter set, creating a problem- and application-specific input file. The cask shielding process is executed with the input by the process manager. This process results in generation of cask-specific dose rates and uncertainty results that are extracted and stored for future use.

2.4.5 Containment Analysis

The cask containment analysis process requires cask type, cask inventory characteristics, and an evaluation date to execute. The process manager aggregates these characteristics from the Unified Database and creates a parameter set. The fuel assembly inventory depletion and decay prerequisite processes are executed for each assembly in a given cask if the results are not available at the evaluation date. The containment template is acquired from the UNF-Templates repository and provided to the TemplateEngine with the parameter set. The TemplateEngine replaces all the variables in the containment template by their respective values and in terms solve the containment analysis equations to calculate the allowable leakage rate. This process generates cask-specific allowable leakage rate results that are extracted from the expanded template and stored in the Unified Database for future use.

CHAPTER

THREE

USER GUIDE

3.1 Graphical User Interface (GUI)

The graphical user interface (GUI) contains the following components (Fig. 3.1, left panel):

- Analysis Tab UNF-ST&DARDS analysis capabilities accessible from this GUI form.
- Exports Tab Commonly requested datasets available in tabular format from this GUI form.
- Pad Maps Tab Geographic Information System (GIS)-Unified Database integrated results navigator.
- SQL Viewer Tab Structured query language input and results form for native readonly Unified Database access.
- Tables Tab Some consolidated Unified Database information in tabular format.

Menu and help buttons are on the top left corner of the GUI. Except for the "change user password" option the other items in the menu drop-down list are under development. "Change user password" can be used to change the password for the Unified Database. The help button provides UNF-ST&DARDS help. In addition to the description of the GUI, the help document provides (1) a description of the configuration file required to run UNF-ST&DARDS, (2) a description of the Unified Database (provided in Section 3.1) with the list and description of all the tables in the Unified Database, and (3) a description of templates (Section 2.2) and the template repository (Appendix A). The analysis tab, pad maps tab, and SQL viewer tab are described below. The exports and tables tabs will be added in a future revision of this manual.

3.1.1 Analysis Tab

The analysis tab is shown in Fig. 3.1. The analysis tab allows for selection of an evaluation type and site or reactor (PWR and boiling water reactor [BWR]) type-specific analysis. Currently the only analysis type activated is the bounding analysis, which refers to analysis



Fig. 3.1: UNF-ST&DARDS GUI.

with conservative reactor irradiation parameters that harden the neutron energy spectrum. Nominal (using nominal reactor irradiation parameters) and detailed (using detailed reactor irradiation parameters) analysis types will be implemented in future.

All analyses can have their execution modes changed to generate the template parameter sets, parameter sets and input, or parameter sets, input, and execution of the analysis application. This can be useful in conducting test runs of a desired analysis and inspecting the analyses parameters or inputs.

When conducting parameter set and input-only execution, the inputs generated may utilize surrogate information, as the actual input would require execution of a prerequisite process. For example, "parameter set" and "input only" execution of canister criticality will generate required isotope discharge and decay parameter sets and inputs. However, it will not execute these inputs, so the canister criticality input being supplemented with surrogate input. This is due to the required isotopic concentration input sections not being generated by the prerequisite decay analyses because no execution of the analysis applications are performed. The start button is used to run the selected analysis type.

Evaluation Type: Isotope

The isotope evaluation type allows for generation of assembly-specific discharged isotopic libraries. These libraries can be used for subsequent assembly isotope decay analysis. These analyses require the precalculated appropriate assembly type interpolation libraries. The analysis scope can be reduced by deselecting the run checkbox for the listed assemblies for which analysis is not desired. Fig. 3.2 shows an isotope analysis option for two selected assemblies (LM01B8 and LM01B9) from the Beaver Valley 1 reactor site. The two assemblies are selected by deselecting all the other assemblies in the run checkbox.

Evaluation Type: Decay

The decay evaluation type is used for the generation of assembly-specific decayed isotopic libraries. These libraries require the corresponding pre-generated discharged isotopic inventory (discussed in Section 4.1.1) libraries. If the discharge libraries do not exist that UNF-ST&DARDS will conduct the pre-requisite discharged isotopic inventory library analyses. The generated assembly-specific decayed isotopic inventory libraries can subsequently be used for criticality, shielding, thermal, and containment analyses. The decay evaluation type requires a specific evaluation date or set of dates (comma delimited) of the format *MM/DD/YYYY*. The analysis scope can be reduced by deselecting the run checkbox for listed assemblies for which analysis is not desired. Results of the decay calculations are stored in the *Decay_Concentration_Summary_bounding* table in the unified database.

Fig. 3.3 shows the decay evaluation type for all the discharged assemblies at Maine Yankee site for two specified decay dates.

000				UNF ST&DARD	S					
Menu Help										
Tables	Step 1 - Evaluation	Туре		Parameters		Step 2	- Site			
Reports	Isotone			Rounding		Sne	cific			
SQL Viewer	Perform disch	area isotona inventory	maberir			0.044	Salart a specific site			
Pad Maps	Perform discharge isotope inventory analysis O Decay						rkansas Nuclear One 1			
Analysis	U Decay						Irkansas Nuclear One 2			
	Perform isoto	pe decay analysis					leaver Valley 1			
	 Canister Criticalit 	ty					Reaver Valley 2			
	Perform canis	ter criticality analysis					Lig Rock Point			
	Cask Thermal						traidwood 1			
	Perform cask	thermal analysis					Braidwood 2			
	O Canister Loading						trowns Ferry 1			
	Perform canis	ter loading evaluation					Browns Ferry 2			
	Canister Design	Rasis							•	
	Berform canie	ter decion basis evoluat				🔾 All	PWR			
	O letteresteries Lib	iter design basis evaluat	ion				Conduct analysis on all PW	R sites		
	Interpolation Lib	rary				() All	BWB			
	Perform inter	polation library calculati	on				Conduct analysis on all BW	P sites		
	Cask Shielding/C	lose				0.41	conduct analysis on an on	in sinces		
	Perform cask	shielding/dose analysis				U All				
	Cask Containmer	nt					Conduct analysis on all sit	25		
	Perform cask	containment analysis								
		paramete	r sets, input, and execution	Decay Time(s) (days) :		Analysis Date(s)		START		
	Details Output									
	Row	Reactor	Assembly	Initial Uranium (kg)	Initial Enrichment (%u235) M	fax Burnup (Mwd/MTHM)	Db ld	Run		
		1 Beaver Valley 1	LM0187	459.969	2.107	20,910.1	219537		٠	
		2 Beaver Valley 1	LM0188	461.467	2.107	17,111.6	219538	×		
		3 Beaver Valley 1	LM0189	459.846	2.107	17,157.1	219524	X		
		4 Beaver Valley 1	LM01BA	462.392	2.107	17,739	219517			
		5 Beaver Valley 1	LM01BB	459.722	2.107	19,515.9	220146			
		6 Beaver Valley 1	LMOIBL	403.108	2.107	29,828.7	220055			
		Requer Valley 1	LMOIBD	404.207	2.107	21,507.0	219521			
		O Requer Valley 1	LMO 1 PE	450.022	2.107	20.251.6	210456			
		10 Reaver Valley 1	LM018C	461.026	2 107	21 907 1	219526			
		11 Beaver Valley 1	LM01BH	459.652	2.107	16,690.9	219539			
		12 Beaver Valley 1	LM01BI	461.282	2.107	29.664.7	220147			
		13 Beaver Valley 1	LM01BK	459.714	2.107	29,650.4	219871			
		14 Beaver Valley 1	LM01BL	462.895	2.107	23,314.7	219599			
		15 Beaver Valley 1	LM018M	460.154	2.107	28,343.8	219729			
		16 Beaver Valley 1	LM018N	460.93	2.107	16,547.9	219532			
		17 Beaver Valley 1	LM018P	459.943	2.107	19,706.7	220148			
		18 Beaver Valley 1	LM01BQ	458.832	2.107	20,028.5	220149			
					0%					

Fig. 3.2: Discharge isotope analysis for two selected assemblies from Beaver Valley 1 reactor site.

000				UNF ST&DARD	IS						
Menu Help											
Tables	Step 1 - Evaluation	Type		Parameters		Step	2 - Site				
Reports	O Isotope			Bounding			Specific				
SQL Viewer	Perform disch	ame isotone inventory a	nalusis	o bounding			Select a specific site				
Pad Maps	Maps: Perform inschange tsvolge internoty analysis Perform isotope decay analysis Canister Criticality Perform canister criticality analysis Canister Loading Perform canister loading evaluation Canister Design Basis Perform canister design basis evaluation Interpolation Library Perform interpolation library calculation Cas Shielding/Dose						Maine Yankee				
Analysis							McGuire 1				
							McGuire 2				
							Millstone 1				
							Millstone 2				
							Millstone 3		-		
							Monticello				
							Morris				
							Nine Mile Point 1		_		
							Alles Adds Dales 3		•		
						• A	II PWR				
							Conduct analysis on all P	WR sites			
						○ A	II BWR				
							Conduct analysis on all B	WR sites			
						○ A	11				
	Perform cask	shielding/dose analysis					Conduct analysis on all s	ites			
	Cask Containmen	nt									
	Perform cask	containment analysis									
		parameter	sate input and execution	The case Time (c) (days) :		Amphasis Data(s) 0/2	0/2015 0/20/2020	STAR	T		
		parameter	sets, input, and execution	ation • Decay Time(s) (days) : Analysis Da			Date(5) (9/30/2015,9/30/2020				
	Details Output										
	Row	Reactor	Assembly	Initial Uranium (kg)	Initial Enrichment (%u235)	Max Burnup (Mwd/MTHM)	Db Id	Run			
		1 Maine Yankee	A001	397	2.01	15,611	245622				
		2 Maine Yankee	A002	396.9	2.01	15,611	245623		-		
		3 Maine Yankee	A004	393.6	2.01	15,951	245624	*			
		4 Maine Yankee	A005	392.3	2.01	15,054	245625	*			
		5 Maine Yankee	A006	393.6	2.01	15,951	245626	*	_		
		6 Maine Yankee	A007	394.5	2.01	11,405	245550	×	_		
		7 Maine Yankee	A008	395.1	2.01	16,511	245627	<u></u>	_		
		8 Maine Yankee	A009	394.1	2.01	16,015	245628	<u> </u>	_		
		9 Maine Yankee	A010	394.2	2.01	16,015	245629				
		10 Maine Yankee	A011	393.7	2.01	16,015	245630	<u> </u>			
		11 Maine Yankee	A012	393.2	2.01	15,940	245631		_		
		12 Maine Tafikee	4014	392.9	2.01	15,394	245632				
		14 Maine Yankee	4015	393.3	2.01	16.015	243033				
		15 Maine Yankee	4016	293.9	2.01	16.015	243034				
		16 Maine Yankee	A017	392.9	2.01	15,951	245635				
		17 Maine Yankee	A018	393.5	2.01	15.951	245637				
		18 Maine Yankee	A019	393.7	2.01	15,394	245638		*		
					0%						
					w/w						

Fig. 3.3: Decay evaluation type for all the discharged assemblies at the Maine Yankee site for two specified dates.
Evaluation Type: Canister Criticality

The canister criticality evaluation type is used for as-loaded, canister-specific criticality safety analysis. These calculations require a significant amount of time to run and potentially large amounts of system resources. This evaluation type requires a specific evaluation date or set of dates (comma delimited) of the format *MM/DD/YYYY*. In addition, the keyword *inservice* can be used to conduct analysis for a canister's inservice date. The analysis scope can be reduced by deselecting the run checkbox for listed canisters for which analysis is not desired.

For criticality analysis, the decayed isotopic inventories for the requested analysis dates are required to be pregenerated for all of the assemblies in the selected canisters. If the decayed isotopic inventories do not exist, UNF-ST&DARDS will conduct the prerequisite assembly-specific decayed isotopic inventory analysis before conducting the criticality analysis. Upon completion of analyses, UNF-ST&DARDS will extract the k-effective and uncertainty values and import them into the Unified Database (*canister_criticality_bounding* table) for future analysis.

Fig. 3.4 presents the criticality analysis type for two specified dates.

			UNF ST	&DARDS			
Menu Help							
Tables Step	1 - Evaluation Type		Parameters		Step 2 - Site		
Reports O Is	sotone		Rounding		Specific		
SQL Viewer	Barform discharpo is a	one inventory analysis	o bounding		Colort a specific site		
Pad Maps	Perform discharge isot	tope inventory analysis			La Crosse		
Analysis	recay				LaSalle County 1		
	Perform isotope decay	analysis			LaSalle County 2		
() C	anister Criticality				Limerick 1		
	Perform canister critica	ality analysis			Limerick 2		-
00	ask Thermal				Maine Yankee		-
	Perform cask thermal a	analysis			McGuire 1		
00	anister Loading				McGuire 2		
	Perform canister loadir	ng evaluation			Millstone 1		_
0.0	anister Design Basis				Millstone 2		•
	Perform canister desig	n basis outlustion			All PWR		
0.1	renorm canister desig	n basis evaluation			Conduct analysis on a	I PWR sites	
	nterpolation Library				All BWR		
	Perform interpolation	library calculation			Conduct analysis on a	I RWR sites	
00	ask Shielding/Dose				O All		
	Perform cask shielding	g/dose analysis			Conduct analysis on al	l class	
00	ask Containment				Conduct analysis on a	li sites	
	Perform cask containm	ent analysis					
	5	parameter sets, input, and exec	ution 💌 Decay Time(s) (days) :	Analysis Date(s)	9/30/2015,9/30/2020	ST	TART
Deta	ils Output						
	Row	Reactor	Canister	Inservice Date	Db Id	Run	
		1 Maine Yankee	TSC-24-NA-TSC-1	Nov 8, 2002	1539	*	^
		2 Maine Yankee	TSC-24-NA-TSC-10	Mar 7, 2003	1548	<u> </u>	
		3 Maine Yankee	TSC-24-NA-TSC-11	Feb 8, 2004	1549		
		4 Maine Yankee	TSC-24-NA-TSC-12	Dec 2, 2002	1550		
		5 Maine Tankee	TSC-24-NA-TSC-15	Pec 38, 2004	1001		
	6 Maine Yankee		TSC-24-NA-TSC-15	Dec 14, 2002	1552		
		8 Maine Yankee	TSC-24-NA-TSC-16	Feb 21 2003	1554		
		9 Maine Yankee	TSC-24-NA-TSC-17	Mar 24, 2003	1555		
		10 Maine Yankee	TSC-24-NA-TSC-18	Oct 26, 2002	1556		
		11 Maine Yankee	TSC-24-NA-TSC-19	Oct 3, 2002	1557		
			100 0 100 100 10	1125 2002	1777	1.1	
		12 Maine Yankee	TSC-24-NA-TSC-2	10125.2005	1540	*	
		12 Maine Yankee 13 Maine Yankee	TSC-24-NA-TSC-2 TSC-24-NA-TSC-20	Sep 13, 2002	1540	× .	
		12 Maine Yankee 13 Maine Yankee 14 Maine Yankee	TSC-24-NA-TSC-2 TSC-24-NA-TSC-20 TSC-24-NA-TSC-21	Sep 13, 2002 Oct 14, 2002	1540 1558 1559	× ×	_
		12 Maine Yankee 13 Maine Yankee 14 Maine Yankee 15 Maine Yankee	TSC-24-NA-TSC-2 TSC-24-NA-TSC-20 TSC-24-NA-TSC-21 TSC-24-NA-TSC-22	Sep 13, 2002 Oct 14, 2002 Sep 21, 2002	1540 1558 1559 1560	* * *	
		12 Maine Yankee 13 Maine Yankee 14 Maine Yankee 15 Maine Yankee 16 Maine Yankee	TSC-24-NA-TSC-2 TSC-24-NA-TSC-20 TSC-24-NA-TSC-21 TSC-24-NA-TSC-22 TSC-24-NA-TSC-23	Jul 25, 2003 Sep 13, 2002 Oct 14, 2002 Sep 21, 2002 Aug 24, 2002	1540 1558 1559 1560 1561	× × ×	
		12 Maine Yankee 13 Maine Yankee 14 Maine Yankee 15 Maine Yankee 16 Maine Yankee 17 Maine Yankee	TSC-24-NA-TSC-2 TSC-24-NA-TSC-20 TSC-24-NA-TSC-20 TSC-24-NA-TSC-21 TSC-24-NA-TSC-22 TSC-24-NA-TSC-23 TSC-24-NA-TSC-24	Jul 25, 2003 Sep 13, 2002 Oct 14, 2002 Sep 21, 2002 Aug 24, 2002 Sep 2, 2002	11540 1558 11559 11560 11561 11562	N N N N N N N N N N N N N N N N N N N	
		12 Maine Yankee 13 Maine Yankee 14 Maine Yankee 15 Maine Yankee 17 Maine Yankee 17 Maine Yankee 18 Maine Yankee	TSC-24-NA-TSC-2 TSC-24-NA-TSC-20 TSC-24-NA-TSC-21 TSC-24-NA-TSC-22 TSC-24-NA-TSC-23 TSC-24-NA-TSC-23 TSC-24-NA-TSC-24 TSC-24-NA-TSC-25	jui 25, 2005 Sep 13, 2002 Oct 14, 2002 Sep 21, 2002 Aug 24, 2002 Sep 2, 2002 Jul 30, 2003	1540 1558 1559 1560 1561 1562 1563	N N N N N N N N N N N N N N N N N N N	_

Fig.	3.4:	Criticality	analysis	type for	all the	canisters	at Maine	Yankee	site for	two	specifie	d
date	s.											

Evaluation Type: Cask Thermal

The thermal evaluation type is used to generate assembly-specific peak and minimum clad temperatures and cask-specific surface temperatures and total decay heat results. These

calculations require a significant amount of time to run. This evaluation type requires a specific evaluation date or set of dates (comma delimited) of the format *MM/DD/YYYY*. In addition, the keyword *inservice* can be used to conduct analysis for a canister's inservice date. The analysis scope can be reduced by deselecting the run checkbox for listed canisters for which analysis is not desired.

For thermal analysis, the decayed isotopic inventories for the requested analysis dates are required to be pregenerated for all the assemblies in the selected canisters. If the decayed isotopic inventories do not exist, UNF-ST&DARDS will conduct the prerequisite assembly-specific decayed isotopic inventory analysis before conducting the thermal analysis. Upon completion of analyses, UNF-ST&DARDS will extract the peak and minimum clad and surface temperature values and import them into the Unified Database (*cask_thermal_bounding* table) for future analysis. In the GUI, the thermal evaluation type appears similar to the criticality evaluation type presented in Fig. 3.4.

Evaluation Type: Canister Design Basis

The canister design basis evaluation type is used for the design basis criticality analysis using a uniform loading map (same burnup and enrichment in all canister locations). A step-by-step description of canister design basis calculation is provided below.

- 1. Select a canister model (see Fig. 7) from the drop-down menu. (Note: Not all canister models have canister input templates, so some may not work).
- 2. Select the assembly type from the drop-down menu to populate the canister.
- 3. Enter a desired 235U enrichment for the assemblies.
- 4. Enter the burnup (megawatt days per metric ton of uranium [MWd/MTU]) to be applied to each assembly.
- 5. Enter a unique canister name.
- 6. Enter a specific power (megawatt per metric ton of uranium [MW/MTU]) to be applied to each assembly.
- 7. Indicate whether it is desired to reuse discharge f71, which will reuse any assemblydischarged isotopic inventory libraries previously generated.
- 8. Indicate if it is desired to reuse decayed values which will reuse any assembly decayed isotopic inventory data previous generated.
- 9. Indicate the canister's inventory count. When this inventory count is specified, the canister inventory table is populated for inspection.
- Specify the decay time(s) (days). If the analysis date(s) is specified, the decay time is computed as the difference (in days) from the day of analysis to the analysis date(s). The analysis date(s) cannot be prior to the day of analysis (i.e., negative decay time).
- 11. Press the START button to conduct the requested analysis.

Note that the inventory can be removed by specifying an inventory count of 0, changing the select parameters, and entering and re-entering inventory count. Fig. 3.5 presents the canister design basis evaluation type.

			UNES	T&DARDS						
Help										
es	Step 1 - Evaluation Type		Parameters			Sten 2 - Site				
rts			Rounding			Spacific				
ewer	Parform discharge isotope inventory	analusis	o bounding			Salart a specific site				
laps	Decay	anarysis				La Crosse				
sis	Perform isotone decay analysis					LaSalle County 1				
	Camintas Calaisaliau					LaSalle County 2				
	Perform socialization and an and an					Limerick 1				
	Coult Thomas					Limerick 2				
	Cask inermal					Maine Yankee				
	Perform cask thermal analysis					McGuire 1				
	O Canister Loading					Millstone 1				
	Perform canister loading evaluation					Millstone 2				
	Canister Design Basis									
	Perform canister design basis evaluat	tion				Conduct analysis on :	all PWP sites			
	 Interpolation Library 					All PWP	an r trik antea			
	Perform interpolation library calculation	on				Conduct analysis on a	all PWP sizes			
	 Cask Shielding/Dose 		All	an own sites						
	Perform cask shielding/dose analysis		- All	all sites						
	renorm casic sincloning/ abse analysis			○ Cask Containment						
	Cask Containment					Conduct analysis on a	an sites			
	Cask Containment Perform cask containment analysis					Conduct analysis on a	an sites			
	Cask Containment Perform cask containment analysis parameter sets,	input, and execution	Decay Time(s) (days) :	An	alysis Date(s) :9/30/2015,9/	30/2020	START			
	Cask Containment Perform cask containment analysis parameter sets, Details Output	input, and execution	Decay Time(s) (days) :	An	alysis Date(s) :9/30/2015,9/	Conduct analysis on a	START			
	Cask Containment Perform cask containment analysis parameter sets, Details Output	input, and execution	Decay Time(s) (days) :	An	alysis Date(s) :9/30/2015;9/	Conduct analysis on a	Burn(dase)			
	Cask Containment Perform cask containment analysis parameter sets, Details Output	input, and execution	Decay Time(s) (days) Type	An Enrichment	alysis Date(s) :9/30/2015.9/ Burnup(MWd/MTU)45.000	Conduct analysis on a 30/2020 Power(MW/MTU) 20	Burn(days)			
	Cask Containment Perform cask containment analysis parameter sets, Output	input, and execution	Decay Time(s) (days) :	Enrichment	alysis Date(s) :9/30/2015,9/ Burnup(MWd/MTU) 45,000 45,000	20/2020 Power(MW/MTU) 30 30	Burn(days) 1,500			
Ĩ	Cask Containment Perform cask containment analysis parameter sets, Details Output Cask	input, and execution	 Decay Time(s) (days) : Type 6151584 8151584 	Enrichment 4.2 4.2	alysis Date(s) :9/30/2015,9/ Burnup(MWd/MTU) 45,000 45,000 45,000	Conduct analysis on a 30/2020 Power(MW/MTU) 30 30 30	Burn(days) 1,500 + 1,500			
Ĩ	Cask Containment Perform cask containment analysis parameter sets, Output	Input, and execution	Decay Time(s) (days) : Type 16151584 16151584 16151584 16151584 16151584	Enrichment 4.2 4.2 4.2 4.2 4.2	alysis Date(s) (9/30/2015,9/ Burnup(MWd/MTU) 45,000 45,000 45,000 45,000	Conduct analysis on a 30/2020 Power(MW/MTU) 30 30 30 30 30	Burn(days) 1,500 1,500 1,500			
đ	Cask Containment Perform cask containment analysis parameter sets. Details Output	input, and execution	 Decay Time(s) (days) : Type 151554 151564 151564 151564 151564 151564 151564 	An Enrichment 4.2 4.2 4.2 4.2 4.2 4.2	alysis Date(s) 9/30/2015.9/ Burnup(MWd/MTU) 45,000 45,000 45,000 45,000	Conduct analysis on a 30/2020 Power(MW/MTU) 30 30 30 30 30 30 30	Burn(days) 1,500 1,500 1,500 1,500			
đ	Cask Containment Perform cask containment analysis parameter sets, Details Output Canister Model MAPC-32 V	Input, and execution	Decay Time(s) (days) : Type 1555564	Enrichment 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2	alysis Date(s) 9/30/2015,9/ Burnup(MWd/MTU) 45,000 45,000 45,000 45,000 45,000	Conduct analysis on a 30/2020 Power(MW/MTU) 30 30 30 30 30 30 30 30 30	Burn(days) 1.500 1.500 1.500 1.500 1.500 1.500 1.500			
đ	Cask Containment Perform cask containment analysis parameter sets. Details Canister Model MPC-32 V Assembly Type	input, and execution	Decay Time(s) (days) : 8151584 8151584 8151584 8151584 8151584 8151584 8151584 8151584 8151584)An Enrichment 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2	alysis Date(s) 9/30/2015.9/ Burnup(MNId/MTU) 45,000 45,000 45,000 45,000 45,000 45,000 45,000	Conduct analysis on a 30/2020 Power(MM/MTU) 30 30 30 30 30 30 30 30 30 30 30 30 30 3	Burn(days) 1,500 1,500 1,500 1,500 1,500 1,500 1,500 1,500			
đ	Cask Containment Perform cask containment analysis parameter sets, Details Output Canister Model MAPC-32 Assembly Type 8151584 • U-235 Enrichment 4.2	input, and execution	Decay Time(s) (days) 151564 1515584	An Enrichment 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2	alysis Date(s) 9/30/2015,9/ Burnup(MMd/MTU) 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000	Conduct analysis on a 30/2020 Power(MW/MTU) 30 30 30 30 30 30 30 30 30 30 30 30 30	Burnidays) 1.500 1.50			
Ż	Cask Containment Perform cask containment analysis parameter sets. Details Output Canister Model MPC-32 V Assembly Type S151584 V U-235 Enrichment 4.2 Burnup OMd/MTUL 45000.0	input, and execution	Decay Time(s) (days) : Type 8151584 8151584 8151584 8151584 8151584 8151584 8151584 8151584 8151584 8151584)An Enrichment 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2	ahysis Date(s) 9/30/2015.9/ Burnup(MMd/MTU) 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000	Conduct analysis on a 30/2020 Power(MM/MTU) 0 0 0 0 0 0 0 0 0 0 0 0 0	Burn(days) 1.500 1.50			
đ	Cask Containment Perform cask containment analysis parameter sets, Details Canister Model MAPC-32 Assembly Type BL515B4 U-235 Enrichment 4,2 Burnup OM0/ MTU, 45000,0 Canister Rame 6 001	input, and execution	Poecay Time(s) (days) 1515584 1515584 1515584 1515584 1515584 1515584 1515584 1515584 1515584 1515584 1515584 1515584	An Enrichment 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2	alysis Date(s) 9/30/2015,9/ Burnup(MMd/MTU) 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000	Conduct analysis on a 30/2020 Power(MW/MTU) 30 30 30 30 30 30 30 30 30 30 30 30 30 3	Burnidayo) 1,500 1,500 1,500 1,500 1,500 1,500 1,500 1,500 1,500 1,500 1,500			
đ	Cask Containment Perform cask containment analysis parameter sets. Canister Model MPC-32 • Assembly Type 1515584 • U-235 Enchment 4.2 Burnup (MWd/MTU) 4.5000.0 Canister Xane 001 Specific Power (MW/MTU) 10	input, and execution view measurements of the second secon	Decay Time(s) (days) Type 1515564 1515564 1515564 1515564 1515564 1515564 1515564 1515564 1515564 1515564 1515564 1515564 1515564) An Enrichment 4,2 4,2 4,2 4,2 4,2 4,2 4,2 4,2 4,2 4,2	alysis Date(s) 9/30/2015.9/ Burnup(MMd/MTU) 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000	Conduct analysis on a 30/2020 Power(SMV/MTU) Power(SMV/MTU)	Burn(days) 1.500 1.500 1.500 1.500 1.500 1.500 1.500 1.500 1.500 1.500 1.500 1.500 1.500 1.500 1.500 1.500 1.500			
đ	Cask Containment Perform cask containment analysis parameter sets, Details Canister Model MMPC-32 Assembly Type BL51584 U-235 Enrichment 4.2 Burnup OM0/ MTUU 45000.0 Canister Rame 6001 Specific Power (MW/MTU) 10 MPC-32,001	input, and execution summer inverses of the second se	Poecay Time(s) (days) 1515564 1515584 1515584 1515584 1515584 1515584 1515584 1515584 1515584 1515584 1515584 1515584 1515584	An Enrichment 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2	alysis Date(s) 9/30/2015,9/ Burnup(MMd/MTU) 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000	Conduct analysis on a 30/2020 Power(MW/MTU) 30 30 30 30 30 30 30 30 30 30 30 30 30 3	Burnidayo) 1,500 1,500 1,500 1,500 1,500 1,500 1,500 1,500 1,500 1,500 1,500 1,500 1,500 1,500 1,500			
Ĩ	Cask Containment Perform cask containment analysis parameter sets, Canister Model MPC-32 • Assembly Type 1515584 • U-235 Enrichment 4.2 Burnup (MWd/MTU) 45000.0 Canister Kane 001 MPC-32.001 Re-Use Discharge f77.1	input, and execution with a second se	Decay Time(s) (days) Type 155564 1655584	An Enrichment 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2	ahysis Date(s) 9/30/2015.9/ Burnup/MMd/MTU) 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000	Conduct analysis on a 30/2020 Power(MW/MTU) 10 30 30 30 30 30 30 30 30 30	Burn(days) 1.500 1.50			
3	Cask Containment Perform cask containment analysis parameter sets. Details Output Canister Model MPC-32 • Assembly Type BISTSB4 • U-235 Enrichment 4.2. Burnup OMM/ MTUU 45000.0 Canister Rame 4001 Specific Power (MW/MTU) 10 MMC-32.001 Re-Use Discharge 77.5 Re-Use Decarded Values 5	input, and execution summer inverses of the second se	Poccay Time(s) (days) Type 81515584 81515584 81515584 81515584 81515584 81515584 81515858 815	An Enrichment 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2	alysis Date(s) 9/30/2015,9/ Burnup(MMd/MTU) 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000	Conduct analysis on a 30/2020 Power(MW/MTU) 30 30 30 30 30 30 30 30 30 30 30 30 30 3	Burn(days) 1,500 1,50			
3	Cask Containment Perform cask containment analysis parameter sets, Canister Model MPC-32 • Assembly Type 1515B4 • U-235 Enchement 4.2 Burnup (MWd (MTU) 4500.0 Canister Name 001 MPC-32.001 Re-Use Decared Yalues Re-Use Decared Yalues	input, and execution with a second se	Decay Time(s) (days) Type 18151584 1815158 1815158 1815158 1815158 1815158 1815158 1815158 18151 1815	An Enrichment 4.2 4.2 4.2 4.2 4.2 4.2 4.2 4.2	alysis Date(s) 9/30/2015.9/ Burnup/MMd/MTU) 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000 45,000	Conduct analysis on a 30/2020 Power(MW/MTU) 30 30 30 30 30 30 30 30 30 30	Burn(days) 1.500 1.50			

Fig. 3.5: Canister design basis evaluation type.

Evaluation Type: Interpolation Library

The interpolation library evaluation type is used for the generation of cross section libraries specific to the assembly type. This analysis can be conducted with the following steps:

- 1. Select the library type from the drop-down list (see Fig. 3.6) of assembly types (Note: Not all assembly types have depletion input templates, so analysis may fail at input generation.)
- 2. Enter the desired enrichment(s).
- 3. Press the START button to commence analysis.

Upon completion of the application, the generated library is copied back to the appropriate arpLib resource directory. Note that the final step in the interpolation library generation is the update of the arplib.txt located at the location that is pointed to by the appropriate arpLib resource. The interpolation library generation requires significant computation resources. It is advised to run only a single process at a time. Fig. 3.6 presents the interpolation library evaluation type as can be seen in the GUI.



Fig. 3.6: Interpolation library evaluation type.

Evaluation Type: Cask Shielding/Dose

The shielding evaluation type is used for the generation of canister-specific shielding analysis for the transportation cask. These calculations require a significant amount of time to run. This evaluation type requires a specific evaluation date or set of dates (comma delimited) of the format *MM/DD/YYYY*. In addition, the keyword *inservice* can be used to conduct analysis for a canister's inservice date. The analysis scope can be reduced by deselecting the run checkbox for listed canisters for which analysis is not desired. For shielding analysis, the decayed isotopic inventories for the requested analysis dates are required to be pregenerated for all the assemblies in the selected canisters. If the decayed isotopic inventories do not exist, UNF-ST&DARDS will conduct the prerequisite assembly-specific decayed isotopic inventory analysis before conducting the thermal analysis. Upon completion of analyses, UNF-ST&DARDS will extract the dose rate values on and near the surface of the transportation cask and import them into the Unified Database (*dose_hac_bounding and dose_nct_bounding* tables) for future analysis. In the GUI, the shielding evaluation type appears similar to the criticality evaluation type presented in Fig. 3.4.

Evaluation Type: Cask Containment

Containment evaluation type is used for canister-specific containment analysis of a transportation package. This evaluation type requires a specific evaluation date or set of dates (comma delimited) of the format *MM/DD/YYYY*. In addition, the keyword *inservice* can be used to conduct analysis for the canister's inservice date. The analysis scope can be reduced by deselecting the run checkbox for listed canisters for which analysis is not desired. For containment analysis, the decayed isotopic inventories for the requested analysis

dates are required to be pregenerated for all the assemblies in the selected canisters. If the decayed isotopic inventories do not exist, UNF-ST&DARDS will conduct the prerequisite assembly-specific decayed isotopic inventory analysis before conducting the containment analysis. Upon completion of analyses, UNF-ST&DARDS will extract the allowable leakage rate values of the transportation cask and import them into the Unified Database (*containment_bounding* tables) for future analysis. In the GUI, the containment evaluation type appears similar to the criticality evaluation type presented in Fig. 3.4.

3.1.2 Pad Maps

The pad maps tab provides information for each site. It provides a GIS-integrated navigation panel, allowing quick site-specific viewing of results. Fig. 3.7 presents the pad map view.



Fig. 3.7: Pad map view.

Typical GIS controls are available in the map view:

- click and drag to pan, or click the pan arrows in the lower left corner,
- double click to center map on clicked point,

- scroll in and out to zoom in and out respectively, or click the + and buttons in the lower left corner,
- hover over a site to display site information in a popup bubble or site information in the information box in lower right corner,
- use the controls in the lower left corner for perspective rotation (left and right) and angle controls, as well as landscape relief increase and decrease, and
- at any point during navigation, reset the view by clicking the reset selection button in the upper right corner.

Nationwide assembly self-protection status (as shown in Fig. 3.8) can also be viewed from the global pad map view by clicking the **Self-Protecting** tab on the upper right corner. The Self-Protecting tab allows user to change the self-protection threshold (dose above which the assemblies are considered as self-protected) and recalculate the self-protection status. The default threshold is set to 100 Rem. Note that in the global pad map view all the other tabs are disabled and they can be only viewed by selecting a site. The following figure presents the nationwide self-protecting status as can be seen by selecting the Self-Protecting tab from the global pad map view.



Fig. 3.8: Self-protecting status of assemblies discharged nationwide.

Clicking the pad maps (site) breadcrumb in the upper right corner provides a list of sites

to choose from. Clicking a site auto-zooms the map perspective to the site's dry storage pad, allowing for inspection of the site's dry storage cask inventory, as shown in Fig. 3.9.



Fig. 3.9: Pad view of a site.

In the site's pad view, hovering the mouse over a cask will provide the cask's identification in a popup bubble. Available details can be viewed by clicking the cask. Clicking the pad maps (cask) breadcrumb in the upper right corner will provide a list of site-specific casks to choose from. The tabs described below are available for each analyzed site.

Cask criticality

The criticality view allows viewing of all site-specific cask criticality results as shown in Fig. 3.10. On the left is the series name listing of all plotted series, with a check box allowing for toggle of the series display. The cask criticality plot is in the center. The cask criticality plot interactions are described in the plots section. The pad map overview is on the right and lists the casks on the pad. Right clicking a row in the pad map overview list provides a context menu which allows for viewing the details of the selected cask or saving the table as a comma separated value (csv) formatted file to a location of the user's choice.



Fig. 3.10: Criticality results for all the canisters of a site.

Cask total decay heat

The total decay heat tab provides a site-wide canister-specific total decay heat view, as shown in Fig. 3.11. On the left is the series name listing of all plotted series, with a check box allowing for toggle of the series display. The total decay heat plot is in the center. The total decay heat plot interactions are described in the plots section. The pad map overview is on the right and lists the casks on the pad. Right clicking a row in the pad map overview list provides a context menu, which allows the user to view the details of the selected cask or to save the table as a comma-separated value (csv) formatted file to a location of the user's choosing.



Fig. 3.11: Total decay heat results for all the canisters of a site.

Cask clad temperatures

The canister cladding temperatures provide site-wide canister-specific cladding temperatures, as shown in Fig. 3.12. The Y-axis displays the cladding temperature (degrees Fahrenheit), and the X-axis shows the evaluation date.



Fig. 3.12: Total Clad temperature results for all the casks of a site.

Cask surface temperatures

The canister surface temperature results provide site-wide canister-specific surface temperatures, as shown in Fig. 3.13. The Y-axis displays the surface temperature (degrees Fahrenheit), and the X-axis shows the evaluation date.

Hypothetical accident condition – 1-meter dose

The canister hypothetical accident condition 1-meter dose provides the site-wide canisterspecific (canister in its designated transportation cask) hypothetical accident condition one-meter dose rates as shown in Fig. 3.14. The Y-axis displays the dose rate (mrem/hr), and the X-axis presents the evaluation date.

Normal condition of transportation – surface dose

The canister normal conditions of transportation surface dose rate results provide site-wide canister-specific (each canister in their designated transportation cask) normal conditions of transportation surface dose rates, as presented in Fig. 3.15. The Y-axis displays the dose rate (mrem/hr), and the X-axis shows the evaluation date.



Fig. 3.13: Surface temperature results for all the casks of a site.

Normal condition of transportation – 2-meter dose

The canister normal conditions of transportation 2-meter dose rate results provide sitewide canister-specific (each canister in their designated transportation cask) normal conditions of transportation 2-meter dose results, as shown in Fig. 3.16. The Y-axis displays the dose rate (mrem/hr), and the X-axis shows the evaluation date.

Normal condition of transportation – personnel boundary dose

The canister normal conditions of transportation personnel boundary dose rate results provide site-wide canister-specific (each canister in its designated transportation cask) normal conditions of transportation personnel boundary dose rates, as shown in Fig. 3.17. The Y-axis displays the dose rates (mrem/hr), and the X-axis shows the evaluation date.

Assembly total decay heat (by burnup)

The assembly total decay heat (grouped by burnup) provides site wide assembly-specific total decay heat as a function of time, as shown in Fig. 3.18. The Y-axis on the left displays the total assembly decay heat (watts/assembly). The color bar on the right displays the burnups, and the X-axis shows the evaluation date.



Fig. 3.14: Hypothetical accident condition 1-meter dose rates for all the canisters at a site in their designated transportation cask.



Fig. 3.15: Normal condition of transport surface dose rates for all the canisters at a site in their designated transportation cask.



Fig. 3.16: Normal condition of transport 2-meter dose rates for all the canisters at a site in their designated transportation cask.



Fig. 3.17: Normal condition of transport personnel boundary dose rates for all the canisters at a site in their designated transportation cask.



Fig. 3.18: The total decay heat as a function of time of all the assemblies at a site grouped by burnup.

Assembly total decay heat (by discharge date)

The assembly total decay heat (grouped by discharge date) provides site-wide assemblyspecific total decay heat as a function of time, as shown in Fig. 3.19. The Y-axis on the left displays the total assembly decay heat (watts/assembly). The color bar on the right displays the discharge dates, and the X-axis shows the evaluation date.

00		UNF ST&DARDS				
Menu Help						
Tables				(
Paparte				Reset Selection	Maine Yankee 👻	[cask] 🔻
COL Manuar						
SQL viewer	Assembly total decay heat (by burnup) Assembly total decay heat (by discharge date)	Self-Protecting	Pad map overview			
Pad Maps	Normal conditions of transportation - allowable release rate	Normal conditions of transportation - allowable leakage rate	Cask serial number Mo	del Vendor Inservice	date Registratio	n Licen
Anaiysis	Hypothetical accident condition – allowable release rate	Hypothetical accident condition – allowable leakage rate	TSC-24-TSC-1 TSC-2	24 NAC Nov 8, 20	02 ML06332023	5 GL 🔺
	Normal conditions of transportation - surface dose Normal conditions of transportation	n - 2-meter dose Normal conditions of transportation - personnel boundary dos	TSC-24-TSC-10 TSC-2	24 NAC Mar 7, 20	03 ML06332023	5 GL
	Pad map Cask criticality Cask total decay heat Cask clad temperatures	Cask surface temperatures Hypothetical accident condition - 1-meter dose	TSC-24-TSC-11 TSC-2	24 NAC Feb 8, 20	14 ML06332023	5 GL
			15C-24-15C-12 15C-2	24 NAC Dec 2, 20	02 ML06332023	5 GL
	Maine Ya	апкее	TSC-24-TSC-14 TSC-	4 NAC FED 10, 2	104 ML06222023	
	Boundi	ng	TSC-24-TSC-15 TSC-2	4 NAC Dec 14 2	102 ML06332023	5 GL
			TSC-24-TSC-16 TSC-2	4 NAC Feb 21, 2	03 ML06332023	5 GL
	1,100		TSC-24-TSC-17 TSC-2	4 NAC Mar 24, 2	003 ML06332023	5 GL =
		1996	TSC-24-TSC-18 TSC-2	24 NAC Oct 26, 20	02 ML06332023	5 GL
	1,050]		TSC-24-TSC-19 TSC-2	24 NAC Oct 3, 201	2 ML06332023	5 GL
		1995	TSC-24-TSC-2 TSC-2	24 NAC Jul 25, 20	03 ML06332023	5 GL
	1,000		TSC-24-TSC-20 TSC-2	24 NAC Sep 13, 2	02 ML06332023	5 GL
		1994	TSC-24-TSC-21 TSC-2	24 NAC Oct 14, 20	02 ML06332023	GL GL
	950 1		TSC-24-TSC-22 TSC-22	4 NAC Sep 21, 2	102 ML06222023	
		1993	TSC-24-TSC-24 TSC-2	4 NAC Sep 2. 20	2 ML06332023	5 GL
	900		TSC-24-TSC-25 TSC-2	4 NAC Jul 30, 20	03 ML06332023	5 GL
		1993	TSC-24-TSC-26 TSC-2	4 NAC Aug 15, 2	003 ML06332023	5 GL
	850		TSC-24-TSC-27 TSC-2	24 NAC Nov 30, 2	003 ML06332023	5 GL
	eno.	199	TSC-24-TSC-28 TSC-2	24 NAC Dec 22, 2	003 ML06332023	5 GL
		100	TSC-24-TSC-29 TSC-2	24 NAC Jan 11, 20	04 ML06332023	5 GL
	750.	4000	TSC-24-TSC-3 TSC-2	24 NAC Oct 29, 20	03 ML06332023	5 GL
	730	1934	TSC 24-TSC-30 TSC-2	24 NAC NOV 18, 2	0.4 ML06332023	5 GL
	700		TSC-24-TSC-31 TSC-2	4 NAC Jan 15, 20	04 ML06332023	
		1989	TSC-24-TSC-32 TSC-2	4 NAC Mar 1, 20	04 ML11033017	B GL
	£ 60.		TSC-24-TSC-34 TSC-2	4 NAC Dec 14, 2	003 ML06332023	5 GL
		1988	TSC-24-TSC-35 TSC-2	4 NAC Nov 9, 20	03 ML06332023	5 GL
	8 600 -		TSC-24-TSC-36 TSC-2	24 NAC Dec 6, 20	03 ML06332023	5 GL
		<u>و</u> 1981	TSC-24-TSC-37 TSC-	24 NAC lun 21 20	03 ML06332023	s ci 💌
	€ 550 B	ę	-			
	t	e, 1986				
	2 500 -	ži stati s		Use a "not" filter		
	<u>e</u>	୍ଥି 198:	Maine Yankee Site			
	2 450 - C	õ	Decommissioning Completed	1		
		1984	• · · · · · · · ·			
	400 -		Maine Yankee	0/1073		
		1983	License Expiration Date: 12/	9/1973 06/1006		
	350 -		Net Electric Capacity: N/A	00/1990		
		1983	Thermal Capacity: N/A			
	300		Reactor Type:PWR			
		198	Nuclear Steam System (NSS	S) CE		
	250 -		Assembly Class: CE 14X14			
		109/	Maine Yankee Wet Storage			
	200 -		-			
		1075	Maine Yankee ISFSI:			
	150 -		NAC-UMS UMS-24:	I		
		407	Loaded Canisters: 64	inc.		
	100	1970	Loaded Assemblies: 1434			
		407	Position: Vertical			
	50 -	197.	Coc: 72-1015			
	······································	1976				
	22222222222222222222222222222222222222					
	Evaluation d	inte 1975				
	Evaluation d	idle	_			

Fig. 3.19: The total decay heat as a function of time of all the assemblies at a site grouped by discharge date.

Normal condition of transport - allowable leakage rate

The canister normal conditions of transportation allowable leakage rate results provide site-wide canister-specific (each canister in their designated transportation cask) normal conditions of transportation allowable leakage rates as shown in Fig. 3.20. The Y-axis displays the allowable leakage rates (cc/sec), and the X-axis shows the evaluation date.

Normal condition of transport - allowable release rate

The canister normal conditions of transportation allowable release rate results provide site-wide canister-specific (each canister in their designated transportation cask) normal



Fig. 3.20: Normal condition of transport containment results (allowable leakage rate) for all the canisters at a site in their designated transportation casks.

conditions of transportation allowable release rates, as shown in Fig. 3.21. The Y-axis displays the allowable leakage rates (curies/sec), and the X-axis shows the evaluation date.



Fig. 3.21: Normal condition of transport containment (allowable release rate) results for all the canisters at a site in their designated transportation casks.

Hypothetical accident condition - allowable leakage rate

The hypothetical accident condition allowable leakage rate results provide site-wide canister-specific (each canister in their designated transportation cask) hypothetical accident condition allowable leakage rates, as shown in Fig. 3.22. The Y-axis displays the allowable leakage rates (cc/sec), and the X-axis shows the evaluation date.

Hypothetical accident condition – allowable release rate

The hypothetical accident condition allowable release rate results provide site-wide canister-specific (each canister in their designated transportation cask) hypothetical accident condition allowable release rates, as shown in Fig. 3.23. The Y-axis displays the allowable leakage rates (curies/sec), and the X-axis shows the evaluation date.



Fig. 3.22: Hypothetical accident condition containment results (allowable leakage rate) for all the canisters at a site in their designated transportation casks.



Fig. 3.23: Hypothetical accident condition containment (allowable release rate) results for all the canisters at a site in their designated transportation casks.

Self-Protecting

Site-wise assembly self-protection status can be viewed (as shown in Fig. 3.24) using the Self-Protecting tab. The Self-Protecting tab allows user to change the self-protection threshold (dose above which the assemblies are considered as self-protected) and recalculate the self-protection status. The default threshold is set to 100 Rem. The following figure presents the Maine Yankee self-protecting status.



Fig. 3.24: Self-protecting status of discharged assemblies at a selected site.

3.1.3 Analysis Model Geometry Viewer

Criticality, thermal, dose, and containment results for a particular canisters/casks can be found by selecting that canister from the pad maps (cask) breadcrumb in the upper right corner.

Fulcrum Editor

If the latest version of SCALE is available in the local machine, then the criticality and the shielding analysis model for the selected canister/cask can be opened in SCALE Fulcrum editor by clicking "view criticality model" and or "view shielding model" (Fig. 3.25).

The Fulcrum editor can be used to plot the analysis model as shown in Fig. 3.26 (Criticality Model). If the dose map is available in the local machine, then the Fulcrum editor can also be used to superimpose a dose map using the mesh tab on the analysis model, as shown in Fig. 3.27. All the available features in the Fulcrum editor can be found in Fulcrum manual

00					UNF ST&DARDS						
Menu Help											
Tables											
Tables							R	eset Selection	Maine Yankee 🔻	TSC-24-TSC-11 -	[assembly] -
Reports											(
SQL Viewer	Containment (allowable le	akage rate) – normal cor	ditions of transportation Con	tainment (allowable release rate)	- normal conditions of transport	rtation	Cask deta	ils			
Pad Maps	Containment (allo	wahle leakane rate) - hv	nothetical accident condition	Containment (all	wahle release rate) - hypothetic	al accident condition	1	Field		Value	
Analysis	Shieldi	ng - hypothetical accide	nt condition	Shieldin	- normal conditions of transpo	artation	Serial num	ber	TS	C-24-TSC-11	
	Canister models	Cask criticality	Total decay heat	Surface temperatures	Clad temperatures	Avial thermal man	Model		TS	C-24	
	cumster models	cusk criticulity	Total decay near	Surface temperatures	ciad temperatures	Axiai thermai map	Cask in-se	ervice date	Fe	b 8, 2004	
							Licensed d	lecay heat load (kW)		
							k-eff at in-	-service date	0.	70553	
							Peak clad	temp at time of	loading ("F) 37	0.142	
							Cask max	surface temp at	time of loading 91	.940	
							Cask loade	ed weight (lbs)	ame or loading (03	.041	
							Cusk loud	eu weigin (ibb)			
							Assembly	position and o	verview		
							Position	Contents In	itial Enrichment (% U-	235) Discharge Burn	up (MWD/MTHM)
							1	G120 3.0	2	30382.0	
							2	G130 3.0	5	30364.0	
							4	T855 3.8	8	27227.0	
							5	T849 3.9	~	27227.0	
							6	0868 3.7		49241.0	
							7	G129 3.0	4	32064.0	
							8	T851 3.8	9	25873.0	
							9	\$434 3.7		33338.0	
							10	\$421 3.7		33338.0	
							11	T853 3.8	8	27227.0	
							12	G134 3.0	3	30364.0	
							13	G135 3.0	3	31996.0	
							14	5010 2.7	1	20307.0	
							16	S431 3.7		33338.0	
							17	T850 3.9		25873.0	
			View	criticality model			18	G128 3.0	3	32135.0	
							19	Q870 3.7		49241.0	
			View	shielding model			20	T848 3.9		26679.0	
							21	T854 3.8	9	25873.0	
							22	Q869 3.6	9	49241.0	
							23	G133 3.0	4	32064.0	
							24	6127 5.0	2	30382.0	
							4				
									view canister	reterence	
							Canned F	uel			
							- 1				

Fig. 3.25: The main display for an individual canister/cask.

The individual cask view (Fig. 3.25) provides results viewing tabs (e.g., cask criticality, surface temperature) similar to those described above for the site view (Fig. 3.9) that provides analysis results of all casks/canisters on the same plot.

Heat Map Viewer

In addition, the individual cask view provides an axial thermal map tab displaying an axial thermal map of the selected cask, as shown in Fig. 3.28. The cask thermal map can be varied axially and as a function of time using the side and bottom scroll bars.

¹ SCALE: A Comprehensive Modeling and Simulation Suite for Nuclear Safety Analysis and Design, ORNL/TM-2005/39, Version 6.1, Oak Ridge National Laboratory, Oak Ridge, Tennessee, June 2011.



Fig. 3.26: Fulcrum editor displaying a criticality model.



Fig. 3.27: Fulcrum editor displaying a superimposed dose map on a shielding model.



Fig. 3.28: Axial thermal map of a selected cask.

3.1.4 Plot Functionality

- **Zoom-in:** A left-click while dragging down and to the right will create a zoom box; releasing the left-click will zoom in.
- **Zoom-reset:** A left-click while dragging up and to the left will initiate the zoom-reset; releasing the left-click will reset the zoom.
- **Plot Properties:** A right-click will display a plot-context menu listing the following:
 - Properties displays configurable plot properties (e.g., title, axis labels, text fonts/size, colors):
 - * Title Tab Allows changing toggle display of title on or off, as well as change of text, font, and color.
 - * Plot Tab Allows changing the domain and range axis label text, font, color, and tick labels, and the plot's appearance (e.g., outline stroke, outline color, background color, and plot orientation).
 - * Legend Tab Allows changing the legend's fonts used to display the plot series names.
 - * Other Tab Allows changing the anti-aliasing rendering used, as well as the

background color.

- Copy copies the plot image to the system's clipboard for subsequent pasting into emails or documents.
- Save as ... Saves the plot as a portable network graphics (PNG) formatted image to the specified location.
- Print... Allows for printing the plot to a selected printer.
- Export... Allows for exporting the plot data as a Microsoft Excel (xslx) formatted file for further data interrogation.

3.1.5 SQL Viewer

The SQL viewer provides direct access to all data stored in the Unified Database. Using SQL select statements, users can precisely define the data they want. The results are displayed in a table from which the user can save the results in a comma-separated value (CSV) file.

As shown in Fig. 3.29, the SQL viewer is divided into left and right panels. The left panel includes a list of table names on top and a tree view of table details at the bottom.

					UNF :	ST&DARDS							
Menu File Optio	ons Help												
Tables	Table serves	1	Eilo 1	-	mbly	v 🗐							
Reports	Table names	34	L'INCI /	asse	inibiy	<u> </u>							
SQL Viewer	admin parameter	Op	en Close	Save	Save a	s Execute qu	ery Cancel o	query					
Results Browser	admin_parameter					· · ·		• •					
Pad Maps	admin_user	sele	ct										
Analysis	admin user role	3554	mbly id										
	analysis job	initia	al uranium	ka.									
	assembly	initia	al enrichm	ent,									
	assembly_array	asse	mbly_type	_id,									
	assembly_code	asse	mbly_cod	_id,									
	assembly_cycle_data	max	_burnup_r	nwd_per_i	mthm,								
	assembly_discharge_cycle	read	tor_id,										
	assembly_location	ansi	_id	1									
	assembly_location_dry	fron	1 assembly	limit 100	100;								
	assembly_location_wet												
	assembly_mox_data		id :	ccambly	id in	itial uranium k	initial enri	chmant	assembly type id	assembly code id	max burgup med per mthm	reactor id	ansi id
	assembly_status	161	647 2/	01	46	1 84700	2 8022000	g	120	assembly_code_id	41919 101	146	ansi_iu
	assembly_type	163	648 24	02	46	5.51	3.8006	0	139		38000.4	146	
	assembly_type_alias	16	649 24	03	46	5 22299	3.8008000	9	139		43271 898	146	
	assembly_type_alias_self_protection	163	650 24	04	464	1 98098	3 8173999	8	139		37646 199	146	
	assembly_type_property	16	651 24	05	46	153201	3 8018999	1	139		40967 101	146	
	assembly_type_specification	163	652 24	06	464	4.69	3.8159000	9	139		43961.101	146	
	authorization	163	653 24	07	46	5.083	3.8138999	9	139		35771.601	146	
	bare_fuel_cask_model	163	654 24	08	46	5.88299	3.8076000	2	139		41890.898	146	
	bare_tuel_cask_model_spec	16	655 2/	09	464	1.03799	3.8022999	8	139		41762.5	146	
	Instrat hina	163	656 ZA	10	46	3.82199	3.8029999	7	139		43283.3	146	
	Table details	163	657 2/	11	464	4.48098	3.8034000	4	139		38015.601	146	
		163	658 2A	.12	466	5.23001	3.8087999	8	139		42902.3	146	
		163	659 2/	13	46	5.36801	3.8062000	3	139		40746.101	146	
		163	660 24	.14	464	4.84698	3.8013000	5	139		42305.398	146	
		163	661 24	15	464	4.613	3.8008000	9	139		35685.601	146	
		163	662 24	.16	463	3.319	3.8029999	7	139		37942.398	146	
		163	663 24	17	46	5.04	3.8194999	7	139		40951.398	146	
		163	664 ZA	18	464	4.57299	3.8192000	4	139		37927.5	146	
		16:	665 24	19	46	5.49899	3.8090000	2	139		42063.5	146	
		163	666 24	20	464	4.19699	3.8046		139		42368.101	146	
		16:	667 2/	21	46	3.36999	3.8059999	9	139		37847.199	146	
		16:	668 ZA	22	46.	2.43798	3.8050999	6	139		40919.3	146	_
		16:	669 24	23	46.	5.38501	3.8071999	ь	139		38114.601	146	
		201 Forr 201 Exe 201 Loa	6/03/14 ning query 6/03/14 cuting se 6/03/14 ding 1000	10:31:42 for assen 10:31:42 lect id, as 10:31:42) rows.	nbly. P	'lease wait. '_id, initial_uran	um_kg, initial_	enrichm	nent, assembly_type_i	d, assembly_code_id,	max_burnup_mwd_per_mthm, r	eactor_id, ansi	_id from asse
			_	_				_					•

Fig. 3.29: UNF-ST&DARDS SQL viewer.

Table names: In the list of table names, right clicking on a table name opens a pop-up menu with options to "select rows" or "show table description." Clicking "select rows" will

generate a SQL query in the query editor window and will then execute the query. Clicking "show table description" will add an entry to the table details tree.

Table details: The table details are displayed as an expandable/collapsible tree. The items in the tree describe the table structure in detail:

- A top-level item shows the table's name. Clicking the icon next to the name (or double clicking the name) will expand the details of the table to reveal a description of the table's contents and a tree item for each column.
- Each column also expands to offer a description and details about the units, datatype, nullability, and uniqueness. This information can be helpful when attempting to write a custom SQL query.

The right panel includes, from top to bottom, a query editor, results table, and message panel.

- Query editor: The query editor panel allows users to write ad hoc queries for retrieving custom data sets from the database. The query editor's menu bar offers several actions:
 - New SQL Tab Creates a blank window for writing a new query.
 - Open Opens an SQL command from a file on your computer.
 - Close Closes the query editor tab.
 - Save Saves changes to the SQL file in the current tab.
 - Save as Saves changes to the SQL file in the current tab with a new name.
 - Execute query Sends the SQL command from the current tab to the database and displays the results. If there is more than one command, the selected command will be executed.
 - Cancel query Attempts to interrupt the currently running query.
- Results table: The query results table displays the results and provides options to "save to CSV," a common format for tabular data and allows users to view the data in most spreadsheet software, like Excel. When saving, specify the filename extension as "csv," as in filename.csv.
- Message panel: The message panel displays information from the SQL Viewer as it communicates with the database.

3.1.6 Reports

The Reports tab provides UNF-ST&DARDS automated Characteristics of Potential Repository Waste report generation capability. The automated report generation capability uses data from the Unified Database to create tables, figures, and other form of data representations. Fig. 3.30 presents the Reports tab. The users can select their favorite Latex

engine using the Latex Executable Directory window (A default is provided with UNF-ST&DARDS package). The users can select the location where the report will be saved from the Document Results Directory window (/Users/kb4/UNF_STANDARDS_reports in the figure below). The Generate PDF button is used to generate the report (in tex and pdf formats). The generated report can be found in the location selected in the Document Results directory window.

Note: The Reports tab is work in progress and mainly shows the automated report generation capability.

_		
In		UNF ST&DARDS
Ψ -	U	NF-ST&DARDS Report Generator
ier is s	.Kull	
	Latex Executable Directory	/usr/local/texlive/2015/bin/x86_64-darwin/pdflatex
	Document Results Directory	Users/kb4/UNF_STANDARDS_reports Browse Generate PDF
	Output Log	
	Save as Clear Search >	
	Found - CrumbereWPRob. Replaced with 87,900 Found - crumbereWRDops 	er2.3.smpl (chapter2.3.smpl potr(chapter2.3.smpl papter2.3.smpl papter2.3.smpl papter2.3.smpl papter2.3.smpl
	" reactorSpecificClasses":12,"numAssemblyClasses":0,"percentFailedFuel":0.955573426114357,"numAssembliesTypes": Successfully Copied JSON Object to File	0,"numberReactors":0)
	SON Object: ("reactors'pecificClasses":12,"humAssemblyClasses".0,"percentFailedFuel":0.955573426114357,"numAsse BELECT 10, tope: (1.1+endor_df, 3.x, 13, y, 43, ype: (2.x+rsion, 5.year, 6, year, 12, width, m, 2.kmg/h, m, 2.kmg/h, whit, co Uvers/hb4/UNF_STADAROD; reports/report (2.35092, 13734/(Charet, 2.7Appendix/ Completed report compliator, Users/kb4/UNF_STANDARDs, reports/report_20150928_113734/(CharacteristicsReport Santing PDF generation.	mbliesTypes10,"numberReactors"0) nrot_rod_m, 12.length_with_bolddown_device_in, t7.assembly_pitch_y_cm, t7.assembly_pitch_y_cm, t8.pitch_cm, t2.total_assembly_weight_kg, t7.initial_uranium_ .tex
	Completed PDF generation: /USers/kb4/UNF_SLANDARDS_reports/report_20150928_113734/CharacteristicsReport.pd	Π. The second

Fig. 3.30: UNF-ST&DARDS report generation tab.

CHAPTER

FOUR

DEVELOPER GUIDE

4.1 Development Environment

Note: This section is under development. The purpose of this section is to provide a quick-start-guide to a new code developer.

UNF-ST&DARDS is comprised of 6 projects:

- 1. The **UNF-Db project** contains the JPA classes to interact with the Unified Database along with some other classes,
- 2. The **EIA859 project** contains the JPA classes to interact with the MySQL GC859 database (direct SQL dump of the MS Access database) and logic to normalize the data into Java objects,
- 3. The **TemplateEngine** expands analysis (e.g., depletion, criticality) templates from **UNF-Templates** with JSON files containing data from the database to create input files for SCALE and the other analysis codes,
- 4. The UNF-Templates contains analysis templates developed by various analysts,
- 5. The **UNF-Db-Model** contains the logic for populating JSON files, running the analysis codes, uploading results to the database, etc.,and
- 6. The **UNF-Db-Viewer** provides a GUI allowing the user to define analysis jobs, submit them for local execution, review results, and interact with the database.

The code repositories are:

UNF-Db: <https://fogbugz.ornl.gov/kiln/Code/Used-Nuclear-Fuel-UNF/Group/UNF-Db>

EIA859: <https://fogbugz.ornl.gov/kiln/Code/Used-Nuclear-Fuel-UNF/Group/EIA859>

TemplateEngine: <https://fogbugz.ornl.gov/kiln/Code/TemplateEngine/Group/TemplateEngine>

UNF-Templates: UNF/Group/UNF-Templates>	<https: code="" fogbugz.ornl.gov="" kiln="" th="" used-nuclear-fuel-<=""></https:>
UNF-Db-Model: UNF/Group/UNF-Db-Model>	<https: code="" fogbugz.ornl.gov="" kiln="" td="" used-nuclear-fuel-<=""></https:>
UNF-Db-Viewer: UNF/Group/UNF-Db-Viewer>	<https: code="" fogbugz.ornl.gov="" kiln="" td="" used-nuclear-fuel-<=""></https:>

4.1.1 Coding Standards

- Write well-documented, orthogonal code that produces the correct results and enables continuous integration with tests.
- Use clear formatting like the default NetBeans code formatter settings.

4.1.2 Test Infrastructure

Whenever code updates are pushed to the central repo, the continuous testing infrastructure runs the regression tests.

Test executions status can be seen @ Jenkins: <http://ci.ornl.gov:8080/view/UNF/>

Test results can be retrieved from **CDash:** <http://ci.ornl.gov/CDash/index.php?project=UNF>

4.1.3 Setting up UNF-ST&DARDS Development Environment

Tools, libraries, and accounts needed:

- 1. Mercurial: <https://www.mercurial-scm.org/>
- 2. NetBeans: <https://netbeans.org/>
- 3. MySQL WorkBench: <https://www.mysql.com/products/workbench/>
- 4. **MySQL Server:** (Optional) If you want to update,insert,delete data during development: <https://dev.mysql.com/downloads/mysql/>
- 5. Linux account on hulk cluster: email Ken Barker <barkerkr@ornl.gov>, cc Kaushik Banerjee <banerjeek@ornl.gov> and John Scaglione <scaglionejm@ornl.gov>
- 6. FogBugz account on fogbugz.ornl.gov: email Tony Walsh <walshtd@ornl.gov>
- 7. CDash account on ci.ornl.gov: email Tony Walsh <walshtd@ornl.gov>
- 8. MySQL account(s) on hulk: Optional: email Paul Miller <millerpp@ornl.gov>
- 9. MacTex: (Optional) If you are working on Latex reporting features, for Mac: https://tug.org/mactex/

10. MikTex: (Optional) If you are working on Latex reporting features, for Windows: http://miktex.org/

Make a parent directory for the projects in your home directory like *C:/Users/<user_id>/NetBeans_Projects*. In subsequent steps we will refer to this directory as the <projects_directory>.

Navigate into the <projects_directory>.

Clone the code from each of the repositories under <projects_directory> using the "hg clone" command.

Clone the Unified Database to a local MySQL Server is optional as mentioned above. The instructions are available at <<u>https://fogbugz.ornl.gov/default.asp?W97</u>>. Do not attempt to use MySQL Workbench to export the dump of the entire Unified Database from hulk:3306; you only need 1 GB ...not the full 30GB database. Additionally, it may be necessary to clone the EIA859 MySQL database depending on the work you are doing. In this case you can use the MySQL Workbench to export the dump. Then restore EIA859 database as detailed in the scripts from <<u>https://fogbugz.ornl.gov/default.asp?W97</u>> with modifications to use the dump_EIA859.sql as the source.

Step 1: Create the projects:

For each of the projects (project setup order: UNF-Db, TemplateEngine, EIA859, UNF-Db-Model, UNF-Db-Viewer):

In NetBeans, left-click the "File" menu. Select "New Project..."

In the dialog, *left-click* Java Project with Existing Sources, then click the *"Next"* button.

In the "New Java Project With Existing Sources" dialog ...

Project Name should be provided according to the repository name, i.e. *"UNF-Db"*.

Project Folder should be <projects_directory>/<project_name>/projects/<user_id>.

Click the *"Next"* button.

Next to the "Source Package Folders:" box, click the "Add Folder..." button.

In the *"Browse Source Packages Folder"* dialog, navigate to *<projects_directory>/<project_name>/src* and click the *"Open"* button.

In the "New Java Project With Existing Sources" dialog, click the "Finish" button.

Step 2: Link the projects and add the libraries:

Project dependency diagram:



For each of the projects (project order: UNF-Db, TemplateEngine, EIA859, UNF-Db-Model, UNF-Db-Viewer):

On the projects navigation panel on the left (NetBeans), *right-click* the project's *"Source Packages"* folder and select *"Properties"*

In the *"Project Properties - < project_name > "* dialog, select *"Libraries"* from the *"Categories:"* panel on the left.

Click the "Add Project..." button for each project listed earlier in the above diagram.

Click the *"Add JAR/Folder"* button and navigate to *<projects_directory>/<project_name>/resource/lib* and select all and click the *"Choose"* button.

The pictures below show any necessary ordering for the projects' libraries. Make sure that the order for your libraries match.

😂 8Model - dist/UNF_Tool.jar .../../resource/lib/dom4j-1.6.1.jar 🍃 8Db - dist/UNF_Db.jar ../../resource/lib/eclipselink-2.3.2.jar 🔜 ../../resource/lib/ext-1.1.1.jar ../../resource/lib/gson-2.2.1-javadoc.jar 🔜 ../../resource/lib/gson-2.2.1.jar ../../resource/lib/hsviewer.jar I./../resource/lib/javax.persistence_2.1.0.v201304241213.jar ../../resource/lib/jcommon-1.0.17.jar ../../resource/lib/jep-2.4.1.jar 同 ../../resource/lib/jfreechart-1.0.14.jar 🗟 ../../resource/lib/jh.jar ../../resource/lib/joda-time-2.3.jar 🔜 ../../resource/lib/junit-4.10.jar ../../resource/lib/kryonet-2.21-all.jar ../../resource/lib/mysql-connector-java-5.1.23-bin.jar I./../resource/lib/opencsv-2.3.jar 🧾 ../../resource/lib/org-netbeans-modules-java-j2seproject-copylibstask.jar ../../resource/lib/org.eclipse.persistence.jpa.jpql_2.5.1.v20130918-f2b9fc5.jar I./../resource/lib/poi-3.8-20120326.jar .../../resource/lib/poi-ooxml-3.8-20120326.jar 🔜 ../../resource/lib/poi-ooxml-schemas-3.8-20120326.jar I./../resource/lib/swing-layout-1.0.4.jar 🔜 ../../resource/lib/xmlbeans-2.3.0.jar I./../resource/lib/zip4j_1.3.2.jar 🔜 ../../resource/lib/worldwind/jogl-all.jar .../../resource/lib/worldwind/gluegen-rt.jar 🥫 ../../resource/lib/worldwind/gdal.jar 🔜 ../../resource/lib/worldwind/gluegen-rt-natives-linux-amd64.jar

Step 3: Configure the resource file

UNF-ST&DARDS reads configuration from the file .unf.rc located in the user's home directory.

4.2 TemplateEngine

The TemplateEngine is a data-driven attribute substitution program. Data are stored in a hierarchical, lightweight format called JavaScript Object Notation (JSON). The TemplateEngine is written in the JavaTM programming language, which means it is cross-platform and should run out of the box. Any question/comment/issue regarding TemplateEngine should be sent to <UNFHelp@ornl.gov> for further assistance. All of the examples provided in this appendix can be found and reproduced from the files located in the TemplateEngine's example directory. A MAC computer is used for the examples in this appendix.

4.2.1 TemplateEngine Requirements

The following system configurations are required for running TemplateEngine:

- Java 1.7 Runtime Environment (JRE 1.7) or newer (<http://www.oracle.com/java>),
- 1 GB RAM or greater, and
- Mercurial 2.5 or newer (<http://mercurial.selenic.com>).

The TemplateEngine is freely available within the Oak Ridge National Laboratory network. Using Mercurial Distributed Version Control System, the TemplateEngine repository can be easily cloned in a local machine as shown in Fig. 4.1. The Mercurial (hg) clone of the repository will download all files required by the TemplateEngine into the TemplateEngine directory. The distribution directory (TemplateEngine/dist) contains the executable java jar file that can now be invoked to expand templates, as shown in Fig. 4.2.

```
$hg clone https://fogbugz.ornl.gov/kiln/Code/TemplateEngine/Group/TemplateEngine
TemplateEngine
requesting all changes
adding changesets
adding manifests
adding file changes
added 47 changesets with 324 changes to 193 files
updating to branch default
142 files updated, 0 files merged, 0 files removed, 0 files unresolved
$
```

Fig. 4.1: Mercurial clone example.

```
$ls TemplateEngine/dist/
README.TXT TemplateEngine.jar lib
$java -jar TemplateEngine/dist/TemplateEngine.jar
***Usage:
    java -jar TemplateEngine.jar /path/to/template [/path/to/json/variables/file]? <-o
outputfile>
$
```

Fig. 4.2: Verification of TemplateEngine execution.

4.2.2 JSON Parameter Set

The TemplateEngine is driven by parameters which are defined in the JSON format. JSON can describe keyed-values, objects, arrays, and compositions of keyed-values, objects and

arrays. The JSON standard is defined at <http://www.json.org>. A parameter in its simplest form is a key-value pair delimited by a colon.

```
key: value
```

The key must be quoted, and is the name of the parameter. The value must be quoted if it is a string value. An object is delimited by left and right curly braces – {}; arrays are delimited by left and right square brackets – [], and commas delimit object and array members.

```
"key1" : value1, "key2" : value2, \dots [value1, value2, \dots]
```

Here, the object contains unique keys, "*key1*" and "*key2*" ... where each key's value could be a primitive value (number, string, etc.) or a nested object or array. Fig. 4.3 depicts an example JSON describing a donut shop's donut options. The document object contains four components, two keyed-values—*type* and *name*—and two arrays—*batter* and *topping*. Batter contains 4 objects, and topping contains 7 objects. Each object in *batter* and *topping* contains two keyed-values, *id* and *type*. This example illustrates the use of keyed-values ("type": "donut", etc.), and arrays ("type", "batter").

```
{
    "type": "donut",
    "name": "Cake",
    "batter":
    [
        { "id": 1001, "type": "Regular" },
        { "id": 1002, "type": "Chocolate" },
        { "id": 1003, "type": "Blueberry" },
        { "id": 1004, "type": "Devil's Food" }
    ],
    "topping":
    [
        { "id": 5001, "type": "Glazed" },
        { "id": 5005, "type": "Sugar" },
        { "id": 5006, "type": "Chocolate with Sprinkles" },
        { "id": 5004, "type": "Chocolate" },
        { "id": 5004, "type": "Chocolate" },
        { "id": 5004, "type": "Chocolate" },
        { "id": 5004, "type": "Maple" }
    ]
}
```

Fig. 4.3: Example donut shop JSON.

4.2.3 TemplateEngine Features

The TemplateEngine feature set is intended to facilitate template expansion involving formatted attribute substitution, expression evaluations, and subtemplate import. The templates can be described with the following components:

- static text,
- attributes,
- indirect attributes,
- scalar expression evaluation,
- iterative expression evaluation,
- subtemplate import,
- conditional action,
- iterative subtemplate import, and
- parameterized subtemplate import.

Template components are described below.

Static text

Static text is the most common template construct. The TemplateEngine conducts a passthrough of the text with no interpretations or evaluations. Fig. 4.4 presents an example of static text in a template.

```
$cat examples/static.tmpl
The brown fox jumped over the lazy dog
$java -jar dist/TemplateEngine.jar examples/static.tmpl > static.txt
$cat static.txt
The brown fox jumped over the lazy dog
$
```

Fig. 4.4: Static text template.

Attributes

Attributes allow for static text to be parameterized to present alternatives. Fig. 4.5 depicts a template containing two attributes, jumper, and what, which allows the attribute values to change, producing different results without modifying the template.

Indirect attributes

Indirect attributes allow attributes to be parameterized, further increasing the flexibility of the template. Arbitrary indirection is allowed. Fig. Fig. 4.6 depicts a template in which the jumper attribute (Fig. 4.5) has been parameterized to incorporate the what attribute, allowing for an indirect attribute lookup based on the value of the what attribute. With this indirect attribute, the change of the what attribute also changes the *jumper[<what>]* attribute.

Fig. 4.7 depicts how changing the value of what from lazy yellow dog to lazy fat cat produces a context change in the result. While the example is simple, the indirect attribute feature is another means of enhancing templates and increasing their flexibility. Fig. 4.7
```
$cat examples/attribute.tmpl
The <jumper> jumped over the <what>
$cat examples/attribute.json
{
    "jumper" : "fluffy brown fox"
    ,"what" : "lazy yellow dog"
}
$java -jar dist/TemplateEngine.jar examples/attribute.tmpl
examples/attribute.json > attribute.txt
Found jumper in examples/attribute.tmpl
Found what in examples/attribute.tmpl
Found <jumper>
---Replaced with fluffy brown fox
Found <what>
---Replaced with lazy yellow dog
$cat attribute.txt
The fluffy brown fox jumped over the lazy yellow dog
$
```

Fig. 4.5: Template with attributes.

```
$cat examples/indirect_attributes.tmpl
The <jumper[<what>]> jumped over the <what>
$cat examples/indirect_attributes.json
    "jumper[lazy yellow dog]" : "fluffy brown fox"
,"jumper[lazy fat cat]" : "cute house mouse"
    ,"what" : "lazy yellow dog"
$java -jar dist/TemplateEngine.jar examples/indirect_attributes.tmpl
examples/indirect_attributes.json > indirect_attributes.txt
Found what in examples/indirect_attributes.tmpl
Found what in examples/indirect_attributes.tmpl
Found <what>
---Replaced with lazy yellow dog
Found <what>
---Replaced with lazy yellow dog
Found <jumper[lazy yellow dog]>
---Replaced with fluffy brown fox
$cat indirect_attributes.txt
The fluffy brown fox jumped over the lazy yellow dog
```

Fig. 4.6: Indirect attribute example 1.

depicts reuse of the *examples/indirect_attributes.tmpl* template with a change in the value of *what*, which changes the result of the template to involve a mouse and a cat instead of a fox and a dog.

```
$cat examples/indirect_attributes1.json
{
    "jumper[lazy yellow dog]" : "fluffy brown fox"
    ,"jumper[lazy fat cat]" : "cute house mouse"
    ,"what" : "lazy fat cat"
}
$java -jar dist/TemplateEngine.jar examples/indirect_attributes.tmpl
examples/indirect_attributes1.json > indirect_attributes1.txt
Found what in examples/indirect_attributes.tmpl
Found what in examples/indirect_attributes.tmpl
Found <what>
---Replaced with lazy fat cat
Found <jumper[lazy fat cat]>
---Replaced with cute house mouse
$cat indirect_attributes1.txt
The cute house mouse jumped over the lazy fat cat
```

Fig. 4.7: Indirect attribute example 2.

Templates may require expressions to be evaluated. The TemplateEngine supports a full suite of mathematical operators and functions. Fig. 4.8 depicts scalar expression evaluations. A scalar expression is formatted as follows:

 $\#evalfmt = format(expression)asnew_attribute\#$

Where the #eval indicates the start of an expression; fmt=format is an optional format specification for the result; the parentheses delimit the expression; the expression is any operation, including a numeric or string-based result; as $new_attribute$ is an optional component that will indicate the result of the expression to be saved for later use by the name $new_attribute$; and the final # closes the expression. On occasion it may be desired for the result of the expression evaluation to remain silent. This can be accomplished with the silent scalar expression using the following format:

```
\# \ evalfmt = format(expression) as new_attribute \#
```

Fig. 4.8 depicts a simple conversion of an attribute in *inches*, *howfar_inches*, to another attribute in *feet*, *howfar_feet*. The first expression evaluated uses the as *new_attribute* feature to store the result of the expression into a re-usable attribute, *howfar_feet*, which is used in the second sentence of the template.

Iterative expression evaluation

The ability to evaluate an expression iteratively as a function of some attribute in the system can be necessary to properly define a template result. The format for an iterative

```
$cat examples/scalar_expr.tmpl
The <jumper> jumped #eval (howfar_inches/12) as howfar_feet# feet over the <what>.
<howfar_feet> feet!
$cat examples/scalar_expr.json
    "jumper" : "fluffy brown fox"
    ,"what" : "lazy yellow dog"
    ,"howfar_inches" : 78
$java -jar dist/TemplateEngine.jar examples/scalar_expr.tmpl examples/scalar_expr.json >
scalar_expr.txt
Found jumper in examples/scalar_expr.tmpl
Found what in examples/scalar_expr.tmpl
Found howfar_feet in examples/scalar_expr.tmpl
Storing evaluated expression howfar_feet as 6.5
Found #eval (howfar_inches/12) as howfar_feet#, replacing with 6.5
Found <jumper>
---Replaced with fluffy brown fox
Found <what>
---Replaced with lazy yellow dog
Found <howfar_feet>
---Replaced with 6.5
$cat scalar_expr.txt
The fluffy brown fox jumped 6.5 feet over the lazy yellow dog. 6.5 feet!
```

Fig. 4.8: Scalar expression example.

expression is as follows:

#funcfmt = formatsep = separator(range)expression#

where *#func* indicates the start of an iterative expression; *fmt=format* is an optional format specification for the results; *sep=separator* is an optional result separator defaulted to a single whitespace; parentheses delimit the *range*; and *expression* is the function to evaluate per iteration. The *range* is of the format:

attribute = start, end

where *attribute* is a new or existing attribute in the template parameters which is iterated from *start* through *end*; and *start* and *end* are numeric values. *Range* can be repeated, whitespace separated, to produce embedded loops.

Fig. 4.9 depicts the squaring of the *i* attribute, as *i* is iterated from 1 to 10 while using the default separator and the comma separator.

Subtemplate import

When dealing with large templates, it is best to split a single file into multiple subtemplates. The TemplateEngine provides a subtemplate import feature that allows import of a file using relative or absolute paths. The format for subtemplate import is as follows:

import file path

\$cat examples/iterative.tmpl
#func (i=1,10) i*i#
#func sep="," (i=1,10) i*i#
\$java -jar dist/TemplateEngine.jar examples/iterative.tmpl > iterative.txt
Found #func (i=1,10) i*i#, replacing with 1.0 4.0 9.0 16.0 25.0 36.0 49.0 64.0 81.0 100.0
Found #func sep="," (i=1,10) i*i#, replacing with
1.0,4.0,9.0,16.0,25.0,36.0,49.0,64.0,81.0,100.0
\$cat iterative.txt
1.0 4.0 9.0 16.0 25.0 36.0 49.0 64.0 81.0 100.0
1.0,4.0,9.0,16.0,25.0,36.0,49.0,64.0,81.0,100.0
\$

Fig. 4.9: Iterative expression example.

Where the *#import* indicates the start of a subtemplate import and *filepath* is an absolute or relative file path. *#import* must start in the first column. Fig. 4.10 depicts the subtemplate import of the *static.tmpl* file. Notice that in the activity message indicating Importing *static.tmpl* using *'null,' 'null'* is an empty parameter set. This will be discussed in parameterized subtemplate import section below. By default, an import of a subtemplate provides the subtemplate access to all attributes in the parent template's attribute set.

```
$cat examples/subtemplate_import.tmpl
#import static.tmpl
$java -jar dist/TemplateEngine.jar examples/subtemplate_import.tmpl >
subtemplate_import.txt
Importing static.tmpl using 'null'
$cat subtemplate_import.txt
The brown fox jumped over the lazy dog
$
```

Fig. 4.10: Subtemplate import example.

Some templates may require that subsections be expanded only when a certain condition applies. The TemplateEngine provides conditional actions to facilitate these types of template expansions. The condition action format is as follows:

```
\begin{array}{l} \# if def condition\\ action\\ \# end if \end{array}
```

where the *#ifdef* or *#ifndef* indicate the start of a conditional action; *condition* is either an attribute name or value, or scalar expression that evaluations to 1 (true), 0 (false), or an attribute name; the *action* is any TemplateEngine feature described in this document; and *#endif* terminates the conditional action. Both *#ifdef* and *#endif* must occur in the first column.

Fig. 4.11 depicts two invocations of the TemplateEngine, exercising two paths through the *examples/conditional_action.tmpl* file. The first invocation determines that the attribute *jumper* exists and then imports the *examples/attribute.tmpl* subtemplate. The second invocation has no JSON parameters specified, so the attribute *jumper* does not exist, and the static text "Nothing jumping today" is printed.

```
$cat examples/conditional_action.tmpl
#ifdef jumper
#import attribute.tmpl
#endif
#ifndef jumper
Nothing jumping today.
#endif
$cat examples/attribute.json
    "jumper" : "fluffy brown fox"
    ,"what" : "lazy yellow dog"
$java -jar dist/TemplateEngine.jar examples/conditional_action.tmpl
examples/attribute.json > condition_action_true.txt
Importing attribute.tmpl using 'null'
Found jumper in examples/attribute.tmpl
Found what in examples/attribute.tmpl
Found <jumper>
---Replaced with fluffy brown fox
Found <what>
---Replaced with lazy yellow dog
$cat condition_action_true.txt
The fluffy brown fox jumped over the lazy yellow dog
$java -jar dist/TemplateEngine.jar examples/conditional_action.tmpl >
condition_action_false.txt
$cat condition_action_false.txt
Nothing jumping today.
$
```

Fig. 4.11: Conditional action example.

Iterative subtemplate import

The ability to import a subtemplate is powerful, but often a subtemplate may need to be imported repeatedly as a function of some attribute. The iterative subtemplate import is one means of repeating an import over a range. The iterative subtemplate import format is as follows:

repeat file path using range

where *#repeat* indicates the start of an iterative subtemplate import; the *filepath* indicates the absolute of relative path to the template to import; using delimits the filepath from the range; and range indicates the repeats to conduct. The *range* is of the format:

$$attribute = start, end$$

where *attribute* is a new or existing attribute in the template parameters, which is iterated from *start* through *end*; and *start* and *end* are numeric values. *Range* can be repeated, whitespace separated, to produce embedded loops.

Fig. 4.12 depicts the iterative import of the *examples/scalar_expr_fmt.tmpl*, which requires three attributes: *jumper*, *howfar_inches*, and *what*. The JSON parameter file, *examples/attribute.json*, provides attributes *jumper* and *what*, and the attribute *howfar_inches* is defined in the range statement of the iterative import where it is iterated from 73 through 75. While the example is simple, the iterative import feature provides a mechanism for generating identifiers, sampling mathematical expressions, and more.

```
$cat examples/iterative_import.tmpl
#repeat examples/scalar_expr_fmt.tmpl using howfar_inches=73,75
$cat examples/scalar_expr_fmt.tmpl
The <jumper> jumped #eval fmt=%.2f (howfar_inches/12) as howfar_feet# feet over the
<what>. <howfar_feet:fmt=%.2f> feet!
$java -jar dist/TemplateEngine.jar examples/iterative_import.tmpl examples/attribute.json
> iterative_import.txt
Repeating examples/scalar_expr_fmt.tmpl using 'howfar_inches=73,75'
<ACTIVITY OMMITTED FOR BREVITY>...
$cat iterative_import.txt
The fluffy brown fox jumped 6.08 feet over the lazy yellow dog. 6.08 feet!
The fluffy brown fox jumped 6.17 feet over the lazy yellow dog. 6.25 feet!
$
```

Fig. 4.12: Iterative import example.

Parameterized subtemplate import

The TemplateEngine works from a data model consisting of a JSON parameter set. JSON describes hierarchical data by using objects and arrays. These hierarchies can be used by corresponding template hierarchies to scope attributes to subtemplates. The parameterized subtemplate import has the following format:

import file path using parameter

where *#import* indicates a subtemplate import; *filepath* indicates the absolute or relative path to the template to import; *using* delimits the *filepath* from the *parameter*; and *parameter* is an attribute expression involving a JSON object or array. When the attribute is an object, the members of that object become available to the subtemplate and are accessible by name. When the attribute is an array, an implicit iterative import is conducted for each member of the array.

Fig. 4.13 depicts a parameterized subtemplate import, which implicitly iterates over both *batter* and *topping* arrays. Each member object of *batter* and *topping* is exposed to the subtemplate *example/parameterized_options.tmpl* where the attributes *id* and *type* are formatted as a bulleted option. Because the TemplateEngine's numerical storage is in floating point values, formatting can be used to remove the decimal portion of the number, creating an integer. This is done by using the format statement *fmt=%.0f*, which states to use 0 decimal places for the floating value.

4.2.4 Attribute and Expression Format

For both attribute and expression evaluation, an optional format specification can be provided as discussed above. Formatting can be used for numeric or string attributes or for expression results. Format is specified in the following format:

$$fmt = \%T.DF$$

where *fmt* indicates a format is to follow; the percent symbol (%) is used to indicate the beginning of the format; T is the optional total space; the decimal (.) and D are the optional decimal space; and F indicates the format type of attribute being formatted.

The total space specification (T) can indicate the alignment. Default alignment is right. A negative sign (-),provides left alignment. A preceding zero (0) indicates that the preceding empty space is padded with zeros. A preceding plus sign (+) indicates to always show the number's sign even when positive.

The decimal space specification (D) can indicate the number of decimal points for numbers or the number of decimal points used as a harsh limiter for strings (can truncate strings). In Fig. 4.14, the first number formatted illustrates decimal truncation, creating an integer-formatted result. The second number formatted illustrates no truncation. The third formatted number indicates six total spaces with two decimal places. The fourth number formatted illustrates padding with zeros. The fifth number formatted illustrates left alignment.

4.2.5 Functions

The scalar and iterative expressions have access to the functions presented in the following tables.

```
$cat examples/parameterized.tmpl
The available selection of <name> <type> is the following:
#eval fmt=%.0f (length(batter)) as batter_count# batters with #eval fmt=%.0f
(length(topping)) as topping_count# toppings.
Toppings (<topping_count>):
#import parameterized_options.tmpl using <topping>
#func sep="-" (i=1,32)"-"#
Batters (<batter_count>):
#import parameterized_options.tmpl using <batter>
$cat examples/parameterized_options.tmpl
    * <type> (<id:fmt=%.0f>)
$cat examples/donut_shop.json
{
    "type": "donut",
    "name": "Cake",
    "batter":
    Γ
         { "id": 1001, "type": "Regular" },
         { "id": 1002, "type": "Chocolate" },
         { "id": 1003, "type": "Blueberry" },
{ "id": 1004, "type": "Devil's Food" }
    ],
    "topping":
    Ε
        { "id": 5001, "type": "None" },
{ "id": 5002, "type": "Glazed" },
{ "id": 5005, "type": "Sugar" },
{ "id": 5007, "type": "Powdered Sugar" },
{ "id": 5006, "type": "Chocolate with Sprinkles" },
{ "id": 5003, "type": "Chocolate" },
{ "id": 5004, "type": "Maple" }
    ]
}
$java -jar dist/TemplateEngine.jar examples/parameterized.tmpl examples/donut_shop.json >
parameterized.txt
<ACTIVITY OMMITTED FOR BREVITY> ...
$cat parameterized.txt
The available selection of Cake donut is the following:
4 batters with 7 toppings.
Toppings (7):
    * None (5001)
    * Glazed (5002)
    * Sugar (5005)
    * Powdered Sugar (5007)
    * Chocolate with Sprinkles (5006)
    * Chocolate (5003)
    * Maple (5004)
                                          Batters (4):
    * Regular (1001)
    * Chocolate (1002)
    * Blueberry (1003)
    * Devil's Food (1004)
$
```

Fig. 4.13: Parameterized subtemplate import example.

\$cat examples/formatting.tmpl
#\eval (pi) as fl##\eval ("string") as str#
|<fl:fmt=%.0f>| |<fl>| |<fl:fmt=%6.2f>| |<fl:fmt=%06.2f>| |<fl:fmt=%-6.02f>|
|<str>| |<str:fmt=%8s>| |<str:fmt=%-8s>| |<str:fmt=%.4s>|
\$java -jar dist/TemplateEngine.jar examples/formatting.tmpl > formatting.txt
<ACTIVITY OMMITTED FOR BREVITY>...
\$cat formatting.txt
|3| |3.141592653589793| | 3.14| |003.14| |3.14 |
|string| | string| |string | |stri|

Fig. 4.14: Several formatting examples.

Description	Function Name
Sine	sin(x)
Cosine	cos(x)
Tangent	tan(x)
Arc Sine	asin(x)
Arc Cosine	acos(x)
Arc Tangent	atan(x)
Arc Tan with 2 parameters	atan2(y, x)
Secant	sec(x)
Cosecant	cosec(x)
Co-tangent	cot(x)
Hyperbolic Sine	sinh(x)
Hyperbolic Cosine	cosh(x)
Hyperbolic Tangent	tanh(x)
Inverse Hyperbolic Sine	asinh(x)
Inverse Hyperbolic Cosine1	acosh(x)
Inverse Hyperbolic Tangent1	atanh(x)

Table 4.1: Trigonometric functions

Table 4.2: Log and exponential functions

Description	Function Name
Natural Logarithm	ln(x)
Logarithm base 10	log(x)
Logarithm base 2	lg(x)
Exponential (e^x)	exp(x)
Power	pow(x)

Table 4.3: Statistical functions

Description	Function Name
Average	avg(x1,x2,x3,)
Minimum	min(x1,x2,x3,)
Maximum	max(x1,x2,x3,)

Table	4.4:	Rounding	functions
-------	------	----------	-----------

Description	Function Name
Round	round(x), round(x, p)
Floor	floor(x)
Ceiling	ceil(x)

Table 4.5: Miscellaneous functions

Description	Function Name
If	if(cond, trueval, falseval)
Str (convert number to string)	str(x)
Absolute Value / Magnitude	abs(x)
Random number (between 0 and 1)	rand()
Modulus	mod(x,y) = x % y
Square Root1	sqrt(x)
Sum	sum(x,y,)
Binomial coefficients	binom(n, i)
Get object or array member by name or index	get(name,index)/
	get(name,child_name)
Get the length of an array or string	length(attribute)
Acquire the left characters in a string	left(string, count)
Acquire the right characters in a string	right(string, count)
Find the index of substr in string	find(string, substr)
Replace all occurrences of X with Y in string	replace(string,x,y)
Determine if a filepath, relative or absolute to the	exists(filepath)
template, exists	
Get the current date milliseconds since January 1, 1970	current_date()
Get the milliseconds since January 1, 1970 and the	date("mm-dd-yyyy")
provided date (mm-dd-yyyy)	

INDEX

A

Assembly decay heat plot, 43

С

cask surface temperature plot, 41 clad temperature plot, 41 criticality plot, 39

D

decay heat plot, 39

F

Fulcrum, 54

Η

HAC allowable leakage rate plot, 51 HAC allowable release rate plot, 51 HAC dose plot, 42 hg mercurial, 66

I

ISFSI view, pad, 37

Μ

map

Pad Map View, 37 mercurial hg, 66

Ν

NCT 2 m dose plot, 42 NCT allowable leakage rate plot, 49 NCT allowable release rate plot, 49 NCT personnel boundary dose plot, 43 NCT surface dose plot, 42

Ρ

pad ISFSI view, 37 plot Assembly decay heat, 43 cask surface temperature, 41 clad temperature, 41 criticality, 39 decay heat, 39 HAC allowable leakage rate, 51 HAC allowable release rate, 51 HAC dose, 42 NCT 2 m dose, 42 NCT allowable leakage rate, 49 NCT allowable release rate, 49 NCT personnel boundary dose, 43 NCT surface dose, 42 Self-protecting (Nationwide), 38 Self-protecting (selected site), 51

S

Self-protecting (Nationwide) plot, 38 Self-protecting (selected site) plot, 51

Т

TemplateEngine attributes, 68 Conditional action, 72 indirect attributes, 68 Iterative expression evaluation, 70 Iterative subtemplate import, 72 Parameterized subtemplate import, 74 Scalar expression evaluation, 70 static text, 68 Subtemplate import, 71

V

view pad ISFSI, 37