

RADIANT Session Changes - December 27-28, 2024

RADIANT Team

December 28, 2024

Contents

RADIANT Platform - Session Changes Report	3
Executive Summary	4
Detailed Changes	5
1. Domain Taxonomy System	5
Purpose	5
Files Created	5
Database Schema	5
Proficiency Dimensions (8)	5
Seeded Taxonomy	6
Detection Algorithm	6
Terminology Signals	6
Model Matching	6
API Endpoints	6
Integration with AGI Brain	7
2. Think Tank Delight System	7
Purpose	7
Files Created	7
Database Tables Created	7
Personality Modes	7
Trigger Types (9)	8
Injection Points (3)	8
Achievements (13)	8
Easter Eggs (10)	8
Sound Themes (5)	9
3. Delight Statistics Dashboard	9
Purpose	9
Files Created	9
Database Tables	9
Trigger Function	9
Statistics Dashboard Features	9
Service Methods Added	10
API Endpoints Added	10
4. Unified Deployment Model	10
Purpose	10
Files Modified	10
Before (Tier-Based)	10
After (Unified)	11
UI Changes	11
Rationale	11

5. Localization System	11
Purpose	11
Files Created/Modified	11
Database Schema	11
Translation Hook	12
Supported Languages	12
6. Windsurf Policy Workflows	12
Purpose	12
Files Created	12
7. Documentation Updates	13
Files Modified	13
New Documentation Sections	13
Summary Statistics	14
Testing Recommendations	15

RADIANT Platform - Session Changes Report

Date: December 27-28, 2024

Duration: 16 hours

Version: 4.18.1 → 4.18.2

Executive Summary

This session implemented five major features and simplified the deployment model:

1. **Domain Taxonomy System** - Hierarchical knowledge domain detection and model matching
 2. **Think Tank Delight System** - Complete personality and engagement system
 3. **Delight Statistics Dashboard** - Persistent analytics with admin UI
 4. **Unified Deployment Model** - Removed tier-based deployment
 5. **Localization System** - i18n infrastructure for Think Tank
-

Detailed Changes

1. Domain Taxonomy System

Purpose

Hierarchical knowledge domain detection system that analyzes user prompts to identify the field, domain, and subspecialty of expertise needed, then matches to optimal AI models based on proficiency scores.

Files Created

File	Purpose	Lines
packages/infrastructure/migrations/045_domain_taxonomy.sql	Initial domain taxonomy	~400
packages/infrastructure/migrations/046_domain_routing_columns.sql	Initial domain routing	~100
packages/infrastructure/migrations/047_learning_domain_columns.sql	Initial learning domain	~50
packages/infrastructure/migrations/048_ethics_bank_domain_columns.sql	Initial ethics bank domain	~50
packages/shared/src/types/domain_taxonomy/types.ts	Domain taxonomy types	~200
packages/infrastructure/lambda/shared/services/domain-taxonomy-service.ts	Domain taxonomy service	~500
packages/infrastructure/lambda/domain-taxonomy/handler.ts	Domain taxonomy handler	~400

Database Schema

```
-- 3-level hierarchy
domain_taxonomy_fields      -- Top level: Science, Humanities, etc.
domain_taxonomy_domains    -- Mid level: Physics, Biology, etc.
domain_taxonomy_subspecialties -- Leaf level: Quantum Physics, etc.

-- Supporting tables
domain_user_selections      -- User's preferred domain settings
domain_detection_feedback   -- AGI learning from corrections
domain_proficiency_mappings -- Model-to-domain scoring
```

Proficiency Dimensions (8)

Dimension	Description	Range
reasoning_depth	Logical reasoning capability	1-10
mathematical_quantitative	Math and numerical analysis	1-10
code_generation	Programming ability	1-10
creative_generative	Creative writing/ideation	1-10
research_synthesis	Research and synthesis	1-10
factual_recall_precision	Accuracy and facts	1-10

Dimension	Description	Range
multi_step_problem_solving	Complex problem solving	1-10
domain_terminology_handling	Technical vocabulary	1-10

Seeded Taxonomy

Fields (12): - Science & Technology - Medicine & Healthcare - Business & Finance - Law & Legal - Arts & Humanities - Education - Engineering - Social Sciences - Environmental Sciences - Mathematics & Statistics - Philosophy & Ethics - Communication & Media

Domains per Field: 5-15 each **Subspecialties per Domain:** 3-10 each **Total Subspecialties:** 500+

Detection Algorithm

```
async detectDomain(prompt: string, options?: {
  include_subspecialties?: boolean;
  min_confidence?: number;
  max_results?: number;
  manual_override?: { field_id?, domain_id?, subspecialty_id? };
}): Promise<DomainDetectionResult>
```

Detection Process: 1. Tokenize and normalize prompt 2. Score each field by keyword matching 3. Score domains within matched fields 4. Score subspecialties by terminology signals 5. Merge proficiency scores from hierarchy 6. Return ranked results with confidence levels

Terminology Signals

Each subspecialty has: - **high_confidence[]** - Terms that strongly indicate this subspecialty - **medium_confidence[]** - Terms that suggest this subspecialty - **exclusionary[]** - Terms that rule out this subspecialty

Model Matching

```
async matchModels(
  proficiencies: ProficiencyScores,
  options?: { max_results?: number; min_score?: number }
): Promise<ModelProficiencyMatch[]>
```

Returns models ranked by: - Match score (0-100) - Dimension-by-dimension scores - Identified strengths/weaknesses - Recommendation flag

API Endpoints

Endpoint	Method	Purpose
/api/v2/domain-taxonomy	GET	Taxonomy overview
/api/v2/domain-taxonomy/full	GET	Complete taxonomy
/api/v2/domain-taxonomy/fields	GET	List fields
/api/v2/domain-taxonomy/fields/:id/fields	GET	Domains for field
/api/v2/domain-taxonomy/domains/:id	GET	Domain details
/api/v2/domain-taxonomy/search	GET	Search taxonomy
/api/v2/domain-taxonomy/detect	POST	Detect from prompt
/api/v2/domain-taxonomy/match-models	POST	Match models
/api/v2/domain-taxonomy/recommend	POST	Recommend orchestration

Endpoint	Method	Purpose
/api/v2/domain-taxonomy/user-select	GET/POST/DELETE	User preferences
/api/v2/domain-taxonomy/feedback	POST	Submit feedback

Integration with AGI Brain

The domain taxonomy integrates with: - **Brain Router** - Uses proficiencies for model selection - **AGI Orchestrator** - Influences orchestration mode - **Neural Engine** - Learns from user feedback - **Delight System** - Domain-specific messages

2. Think Tank Delight System

Purpose

The Delight System adds personality, humor, and engaging feedback to Think Tank AI interactions, making the experience more delightful and human.

Files Created

File	Purpose	Lines
packages/infrastructure/migrations/075-add-delight-system-messages.js	Database migrations for delight system messages, achievements, easter eggs	~400
packages/shared/src/types/RelayScriptTypes.ts	Relay script type definitions	~200
packages/infrastructure/lambda/higher/services/delight.service.ts	Core delight service logic	~900
packages/infrastructure/lambda/shared/generators/delight-orchestration.service.ts	Delight orchestration logic	~400
packages/infrastructure/lambda/shared/services/delight-events.service.ts	Delight event handling logic	~500
packages/infrastructure/lambda/delight/handler.ts	Delight system lambda handler	~500
apps/admin-dashboard/app/(dashboard)/thinktank/delight/page.tsx	Admin dashboard page for delight system	~350

Database Tables Created

```
-- Core tables
delight_categories      -- 10 categories (domain_loading, time_aware, etc.)
delight_messages        -- Messages with targeting options
delight_achievements    -- 13 predefined achievements
delight_user_achievements -- User unlock tracking
delight_easter_eggs      -- 10 hidden features
delight_easter_egg_discoveries -- User discovery tracking
delight_sound_themes     -- 5 sound themes
delight_user_preferences -- Per-user settings
delight_event_log        -- Activity tracking
```

Personality Modes

Mode	Intensity	Use Case
professional	1-3	Business/enterprise users
subtle	4-5	Default experience
expressive	6-7	Engaged users

Mode	Intensity	Use Case
playful	8-10	Power users, fun seekers

Trigger Types (9)

1. **domain_loading** - “Consulting the quantum realm...”
2. **domain_transition** - “Shifting gears to biology...”
3. **time_aware** - “Burning the midnight tokens”
4. **model_dynamics** - “Consensus forming across models...”
5. **complexity_signals** - “This one’s a puzzle...”
6. **synthesis_quality** - “High confidence synthesis achieved”
7. **achievement** - “Achievement unlocked: Domain Explorer!”
8. **wellbeing** - “You’ve been at it for 2 hours. Break time?”
9. **easter_egg** - Hidden surprise triggers

Injection Points (3)

Point	When	Example Messages
pre_execution	Before AI generates	Domain loading, time awareness
during_execution	While generating	Model dynamics, step progress
post_execution	After completion	Synthesis quality, achievements

Achievements (13)

ID	Name	Threshold	Points
first_chat	First Steps	1 chat	10
power_user	Power User	100 chats	100
domain_explorer	Domain Explorer	10 domains	50
week_warrior	Week Warrior	7-day streak	75
night_owl	Night Owl	10 late sessions	25
early_bird	Early Bird	10 morning sessions	25
renaissance_mind	Renaissance Mind	50 domains	200
monthly_mind	Monthly Mind	30-day streak	150
model_master	Model Master	20 models used	75
deep_thinker	Deep Thinker	50 extended thinking	100
collaborator	Collaborator	10 shared chats	50
artifact_artist	Artifact Artist	25 artifacts	75
feedback_friend	Feedback Friend	20 feedback given	50

Easter Eggs (10)

ID	Trigger	Effect
konami_code	↑↑↓↓←→←→BA	Rainbow mode
disco_mode	“disco” typed	Disco lights
matrix_mode	“matrix” typed	Matrix rain
pirate_speak	“arrr” typed	Pirate responses
shakespeare	“forsooth” typed	Shakespearean AI
time_traveler	Midnight exactly	Time travel message

ID	Trigger	Effect
lucky_seven	7:07:07 time	Lucky message
palindrome	Palindrome date	Special message
birthday	User's birthday	Celebration
anniversary	1-year anniversary	Thank you

Sound Themes (5)

Theme	Style	Sounds
default	Pleasant chimes	thinking, success, achievement
mission_control	NASA-inspired	beeps, confirmations
library	Quiet, bookish	page turns, soft clicks
workshop	Mechanical	tool sounds, clicks
emissions	Tesla-style	electric hums

3. Delight Statistics Dashboard

Purpose

Persistent analytics for tracking Delight System usage, engagement, and performance.

Files Created

File	Purpose	Lines
packages/infrastructure/migrations/016_create_delight_stats.sql	Creates the database table for statistics	~250
apps/admin-dashboard/app/javascripts/link/delight/statistics/page.tsx	Dashboard page for tracking link/delight/statistics	~350
apps/admin-dashboard/app/javascripts/link/delight/statistics/route.ts	API endpoint for the dashboard	~100

Database Tables

```

delight_daily_stats      -- Daily aggregated statistics
delight_message_stats    -- Per-message performance
delight_achievement_stats -- Achievement analytics
delight_easter_egg_stats  -- Easter egg discovery rates
delight_weekly_trends     -- 12-week trend data
delight_user_engagement  -- User leaderboard data

```

Trigger Function

```

CREATE FUNCTION update_delight_daily_stats()
-- Automatically updates statistics on every delight_event_log insert
-- Aggregates: messages_shown, achievements_unlocked, easter_eggs_found,
--             sounds_played, unique_users, avg_intensity

```

Statistics Dashboard Features

Section	Metrics
Overview Cards	Total messages, achievements, easter eggs, active users
Weekly Trends	12-week chart of all activity types
Top Messages	Most-shown messages with engagement rates
Achievement Stats	Unlock rates, time-to-unlock, rarity distribution
Easter Egg Stats	Discovery rates by egg, trending discoveries
User Engagement	Leaderboard by achievement points

Service Methods Added

```
// In delight.service.ts
async getDetailedStatistics(tenantId: string): Promise<DelightStatistics>
async getDailyStats(tenantId: string, days: number): Promise<DailyStats[]>
async getTopMessages(tenantId: string, limit: number): Promise<MessageStats[]>
async getAchievementStats(tenantId: string): Promise<AchievementStats[]>
async getEasterEggStats(tenantId: string): Promise<EasterEggStats[]>
async getWeeklyTrends(tenantId: string, weeks: number): Promise<WeeklyTrend[]>
async getUserEngagement(tenantId: string, limit: number): Promise<UserEngagement[]>
```

API Endpoints Added

Endpoint	Method	Response
/api/admin/delight/statistics	GET	Full statistics object
/api/admin/delight/user-engagement	GET	User leaderboard

4. Unified Deployment Model

Purpose

Simplified deployment by removing tier-based infrastructure selection. All features are now available in every deployment; licensing restrictions will be handled at the application level.

Files Modified

File	Change
apps/swift-deployer/Sources/RadiantDeployer/Views/DeployView.swift	Is swiftstate variable, tier pickers
apps/swift-deployer/Sources/RadiantDeployer/Views/DeploymentParametersEditorView.swift	DeploymentParametersEditorView
apps/swift-deployer/Sources/RadiantDeployer/Services/CDKService.swift	CDKService.swift deploy function

Before (Tier-Based)

```
// Old deployment with tier selection
func deploy(
    appId: String,
    environment: String,
    tier: Int, // ← REMOVED
    credentials: CredentialSet,
```

```

        progressHandler: @escaping (String) -> Void
    ) async throws -> DeploymentOutputs?

// CDK context
"--context", "tier=\(tier)" // ← REMOVED

```

After (Unified)

```

// New unified deployment
func deploy(
    appId: String,
    environment: String,
    credentials: CredentialSet,
    progressHandler: @escaping (String) -> Void
) async throws -> DeploymentOutputs?

```

UI Changes

Removed from DeployView: - @State private var selectedTier: TierLevel = .seed - Tier selection picker section - TierPickerNew component (80 lines) - TierPicker component (40 lines) - Tier references in deployment log messages

Removed from ParameterEditorView: - Tier picker section - Tier-based feature disabling (Multi-AZ, Self-hosted, Multi-region) - Tier description in parameter source text

Rationale

Old Model	New Model
5 tiers with different infrastructure	Single unified infrastructure
Features gated by tier	Features gated by license
Complex deployment decisions	Simple deployment
Infrastructure-level restrictions	Application-level restrictions

5. Localization System

Purpose

Internationalization infrastructure for Think Tank UI strings.

Files Created/Modified

File	Purpose
packages/infrastructure/migrations/0741_create_string_registry_table.sql	UI string registry table migrations
apps/admin-dashboard/hooks/useTranslations.ts	Frontend translation hook
Settings UI	Language selector component

Database Schema

```

-- UI string registry
CREATE TABLE ui_strings (

```

```

    id UUID PRIMARY KEY,
    namespace VARCHAR(50),           -- 'common', 'chat', 'settings'
    key VARCHAR(255),                -- 'welcome_message'
    default_value TEXT,              -- English default
    context TEXT                     -- Usage context
);

-- Translations
CREATE TABLE ui_translations (
    id UUID PRIMARY KEY,
    string_id UUID REFERENCES ui_strings(id),
    language_code VARCHAR(10),       -- 'es', 'fr', 'de', 'ja'
    translation TEXT,
    is_verified BOOLEAN
);

```

Translation Hook

```

function useTranslation(namespace: string = 'common') {
    const t = (key: string, params?: Record<string, string>) => string;
    const language: string;
    const setLanguage: (lang: string) => void;
    const isRTL: boolean;
}

```

Supported Languages

Code	Language	Status
en	English	Default
es	Spanish	Seeded
fr	French	Seeded
de	German	Seeded
ja	Japanese	Seeded

6. Windsurf Policy Workflows

Purpose

Development policy enforcement workflows for the Windsurf/Cascade AI agent.

Files Created

File	Policy
.windsurf/workflows/no-hardcoded-ui-text.md	All UI strings must be localized
.windsurf/workflows/no-mock-data.md	No mock data in production code
.windsurf/workflows/no-stubs.md	No stub implementations in production
.windsurf/workflows/hipaa-phi-sanitization.md	HIPAA/PHI input sanitization

7. Documentation Updates

Files Modified

File	Changes
CHANGELOG.md	Added v4.18.2 section with all changes
docs/DEPLOYMENT-GUIDE.md	Removed tier references, added unified deployment note
docs/THINK-TANK-USER-GUIDE.md	Added Section 13: Delight System (user-facing)
docs/RADIANT-ADMIN-GUIDE.md	Added Section 20: Delight System Administration
docs/generate-pdfs.sh	Added complete documentation PDF generation

New Documentation Sections

Think Tank User Guide - Section 13: - What is Delight? - Personality Modes - Achievements - Easter Eggs - Sound Effects - Customization

Admin Guide - Section 20: - Accessing Delight Admin - Dashboard Overview - Managing Categories - Managing Messages - Statistics Dashboard - Managing Achievements - Managing Easter Eggs - API End-points

Summary Statistics

Metric	Count
Files Created	25+
Files Modified	15+
Database Migrations	7 (045-048 Domain, 074-076 Delight/i18n)
Database Tables	25+
API Endpoints	25+
UI Components	8+
Lines of Code	~5,000+
Documentation Pages	150+

Testing Recommendations

1. **Domain Taxonomy**
 - Test domain detection with various prompts
 - Verify proficiency score merging
 - Test model matching algorithm
 - Validate user selection persistence
 - Test feedback submission and learning
2. **Delight System**
 - Test message triggering at each injection point
 - Verify achievement unlocking logic
 - Test easter egg activation
 - Verify sound playback
3. **Statistics Dashboard**
 - Verify data aggregation triggers
 - Test weekly trend calculations
 - Validate leaderboard sorting
4. **Unified Deployment**
 - Run Swift Deployer and verify no tier UI
 - Test deployment without tier parameter
 - Verify CDK deployment succeeds
5. **Localization**
 - Test language switching
 - Verify translations load correctly
 - Test RTL language detection

Report generated: December 28, 2024 Session duration: 16 hours Version: 4.18.2