CONCEPTS GUIDE

SL 5N

USAREUR-AF

CONCEPTS GUIDE — SL 5N COMPANION — ADVANCED UI/UX DESIGNER · MAVEN SMART SYSTEM (MSS)

Specialist Course Manual

HEADQUARTERS
UNITED STATES ARMY EUROPE AND AFRICA
(USAREUR-AF)
Wiesbaden, Germany

26 MARCH 2026

DRAFT — UNOFFICIAL — NOT FOR OPERATIONAL USE

CONCEPTS GUIDE — SL 5N COMPANION — ADVANCED UI/UX DESIGNER · MAVEN SMART SYSTEM (MSS)

Forward: SL 5N moves from designing applications to designing the system that produces applications. The advanced Designer's output is not a single interface — it is the standards, patterns, and processes that make every interface better. **Purpose:** Develops mental models required to lead design at enterprise scale on MSS — design systems, DDIL patterns, coalition considerations, and design operations. Read before beginning SL 5N task instruction. *HQ USAREUR-AF · v1.0 · 2026 · DISTRIB: USG only*

SECTION 1 — FROM APPLICATION DESIGN TO DESIGN SYSTEMS

1-1. The Core Transition

BLUF: A design system is a product that serves other products. Your users are designers and developers, not end users.

At SL 4N, the Designer builds one application at a time. The quality of that application depends on the individual Designer's skill, judgment, and consistency. That model does not scale. When the MSS portfolio grows to 20, 50, or 100 applications, individual heroics cannot sustain quality — systemic solutions can.

A SL 5N Designer who continues designing individual applications is underperforming. The advanced Designer's job is to build the infrastructure — component libraries, design tokens, pattern documentation, governance processes — that makes every SL 4N Designer more effective. This is the same shift an advanced SWE makes from writing code to building the systems that make code better.

1-2. The Scale Shift

SL 4N (Application Design)	SL 5N (Design Systems)
"How should this dashboard look?"	"How should all dashboards look?"
"What color means 'ready'?"	"What color system ensures consistency across 50 applications?"
"Is this form accessible?"	"How do we ensure every form across the portfolio is accessible?"
"I tested with 5 users"	"How do we systematize research so findings benefit all teams?"

SL 4N (Application Design)	SL 5N (Design Systems)
"This pattern works for my app"	"This pattern is documented, tested, and available to every team"

1-3. The Design System Product Mindset

The design system has its own backlog, roadmap, and release cycle. Component requests from application teams are feature requests. Breaking changes require migration guides. Deprecating a component requires a sunset period. Treat the design system with the same rigor SL 4L applies to shared code libraries.

Component governance model:

Decision	Who Decides	Criteria
Add a new component to the library	Design system team (2+ designers)	Used by 3+ applications; documented; accessible; responsive
Modify an existing component	Design system team + affected consumers	Non-breaking: ship. Breaking: deprecation window required
Approve a pattern deviation	Lead Designer + requesting Designer	Standard pattern genuinely does not fit; exception documented
Deprecate a component	Design system team	Replacement available; migration guide published; sunset period set

The versioning discipline: A design system is a dependency for every MSS application. Unversioned component changes break downstream applications the same way unversioned API changes break consumers. Apply semantic versioning: MAJOR for breaking changes, MINOR for additions, PATCH for fixes.

SECTION 2 — DDIL DESIGN THINKING

2-1. The Operating Reality

BLUF: DDIL is not an edge case — it is a primary operating condition in the USAREUR-AF AOR. Design for disconnection first; connectivity is the bonus.

Commercial design assumes connectivity. Military design cannot. Every MSS application will operate at some point under denied, disrupted, intermittent, or limited bandwidth conditions. The Designer who treats DDIL as an edge case will produce applications that fail when they are needed most — during

operations, not garrison.

2-2. The Offline-First Mental Model

```
DESIGN FOR:                    NOT FOR:
No network → Cached data       Always connected → Live data
Slow network → Progressive     Fast network → Load everything
  loading                        at once
Intermittent → Queue and sync  Stable → Immediate server
                                 response
```

Design implications by DDIL tier:

Tier	Network Condition	Design Response
Full connectivity	Normal bandwidth, low latency	Full functionality; real-time data
Degraded	Reduced bandwidth, higher latency	Reduce payload; defer non-critical updates; show loading indicators with time estimates
Intermittent	Connection drops and reconnects	Queue actions locally; sync on reconnect; show connection status; indicate data freshness
Disconnected	No connectivity	Read-only cached data; indicate staleness prominently; queue writes for later sync

2-3. Data Freshness as a Design Variable

In a connected commercial app, data is assumed current. In MSS under DDIL, every data element has an age — and that age is operationally relevant. A readiness report from 10 minutes ago is useful. From 4 hours ago, it requires a caveat. From 24 hours ago, it may be misleading.

The Designer must encode these freshness states visually so the user never makes a decision on unknowingly stale data. This requires three elements on every data display:

1. **Last updated timestamp** — when was this data last confirmed current?
2. **Staleness threshold indicator** — visual cue when data exceeds its freshness window (e.g., readiness data >4 hours old gets amber border)
3. **Source status** — is the upstream data source currently reachable?

The cardinal DDIL design error: Displaying stale data with no staleness indicator. This creates false confidence — the user believes they are making a decision on current information when they are not. A blank screen is safer than an unknowingly stale screen.

SECTION 3 — CROSS-DOMAIN AND COALITION MENTAL MODELS

3-1. Classification as a Design Constraint

BLUF: Classification boundaries are security boundaries. The Designer cannot make them invisible — but can make them unambiguous.

The user must always know, without doubt, what classification level they are operating at and what classification the data they are viewing carries. This is not a UX problem to "solve" by making it seamless — it is a security requirement to make explicit.

Cross-domain design requirements: - Classification banners visible at ALL times — never hidden by scroll, modal, or overlay - Color-coded banners per IC/DoD standard with redundant text labels (color alone is not sufficient — accessibility and screen glare degrade color perception) - Data from different classification levels NEVER displayed on the same screen without explicit domain separation and ISSM-approved design - Session transition between classification levels requires deliberate user action (not automatic) - Print output includes classification markings on every page

3-2. The Coalition Design Challenge

When 20 nations use the same interface, design for the lowest common denominator of shared understanding. Military terminology varies across nations. Abbreviations that are obvious to a US Soldier may be meaningless to a German Soldat or Polish Żołnierz.

Coalition UI considerations:

Consideration	Design Response
Language	English as primary; critical labels and status indicators designed for non-native English readers (simple vocabulary, no idioms, no abbreviations without expansion)
Date/time format	DTG standard for military use; ISO 8601 as fallback; never MM/DD/YYYY (ambiguous internationally)
Units of measure	Metric primary for coalition contexts; dual display where required
Releasability markings	REL TO markings displayed alongside classification; filter controls enforce releasability
Cultural conventions	Left-to-right layout assumption documented; color associations verified across partner nation conventions

3-3. Cross-Domain Workflow Design

The most dangerous design error in cross-domain work: making classification transitions feel routine. Every classification boundary crossing must require deliberate action, explicit confirmation, and visual reinforcement. Designers who optimize for speed across classification boundaries are optimizing for spillage.

Vignette — USAREUR-AF Coalition Dashboard: An MSS dashboard displays readiness data at RELIDO level for coalition view and at NOFORN level for US-only analysis. The Designer must ensure that switching between views requires an explicit button press with a confirmation dialog stating the new classification level. Auto-switching between tabs or views that cross classification boundaries is never acceptable.

SECTION 4 — DESIGNOPS AS FORCE MULTIPLIER

4-1. What DesignOps Is

BLUF: DesignOps is to design what DevOps is to development — the operational practices that let design scale beyond individual heroics.

A single talented Designer can produce excellent applications. Two talented Designers can produce excellent but inconsistent applications. Ten Designers without DesignOps produce chaos. DesignOps is the set of processes, tools, and governance that ensures consistent quality as the design team and application portfolio grow.

4-2. DesignOps Functions

Function	Purpose	Without DesignOps
Design system maintenance	Keep components current, documented, accessible	Components drift; each app reinvents patterns
Research repository	Prevent duplicate research; share insights across teams	Same user groups interviewed repeatedly; insights lost
Design review governance	Ensure consistency and accessibility across portfolio	Quality varies by individual designer skill
Tooling and templates	Reduce setup time; enforce standards from the start	Every project starts from scratch
Onboarding	Get new designers productive quickly	Months of ramp-up; tribal knowledge

4-3. The Research Repository

User research is expensive (operational access, security clearance requirements, user availability). Every research finding should be documented, tagged, and searchable so that future teams can build on existing knowledge rather than re-conducting the same studies.

Research repository structure: - **Study records:** Who was studied, when, where, methodology, key findings - **Insight library:** Validated insights tagged by user role, WFF track, application domain - **Persona library:** Maintained set of user personas updated with each research cycle - **Recommendation tracker:** Design recommendations with implementation status

The rotation problem: Military units rotate. The user base changes every 2-3 years. Research findings from one rotation may not apply to the next — but the underlying tasks and workflows are stable. Tag research by task, not just by unit. Task-based insights survive rotations; unit-specific preferences may not.

4-4. Design Quality Metrics

Metric	What It Measures	Target
Accessibility compliance rate	% of applications meeting WCAG 2.1 AA	100%
Design system adoption	% of components using design system tokens/components	>90%
Usability test cadence	% of applications tested with representative users in past 6 months	100%
Design-to-deploy fidelity	% of implemented features matching design specification	>95%
Time to first design	Days from requirement to first design review	<5 business days
Pattern reuse rate	% of new designs using existing patterns vs. creating new ones	>70%

SECTION 5 — ACCESSIBILITY AT ENTERPRISE SCALE

5-1. Why Accessibility Is Non-Negotiable

BLUF: Accessibility is a legal requirement, an operational necessity, and a design quality indicator. An application that is not accessible is not finished.

MSS users operate in environments with screen glare, dim lighting, stress, fatigue, and sometimes injury. Accessibility is not about accommodating a minority — it is about ensuring every user can operate the system under degraded conditions that are the norm in operational environments.

5-2. Automated vs. Manual Testing

Can Automate	Must Test Manually
Color contrast ratios	Logical reading order for screen readers
Missing alt text	Meaningfulness of alt text content
Missing form labels	Clarity of error messages
Keyboard focus indicators present	Keyboard navigation flow is logical
ARIA roles present	ARIA roles are correct for the interaction
Touch target size	Touch target placement makes sense in context

Automation catches approximately 30-40% of accessibility issues. The remaining 60-70% require human evaluation. Design the automated suite first — it catches regressions at zero marginal cost. Then schedule manual testing on a cadence that covers the portfolio.

5-3. Remediation Prioritization

Priority	Criteria	Response
P0 — Blocker	User cannot complete the primary task at all	Fix before next deployment
P1 — Critical	User can complete the task but with significant difficulty	Fix within current sprint
P2 — Major	User experience is degraded but task is completable	Schedule in next sprint
P3 — Minor	Cosmetic or minor friction; does not affect task completion	Backlog; address during next design system update

SECTION 6 — ADVANCED FAILURE MODES: WHAT SL 5N DESIGNERS GET WRONG

6-1. Overview

BLUF: Senior Designers make different mistakes than junior Designers. They are less likely to produce poor layouts and more likely to over-engineer systems, ignore operational context, or build design infrastructure that no one adopts.

6-2. Summary Table — Advanced Failure Modes

Failure Mode	Description	Diagnostic Question
Over-engineering the design system	Building components for hypothetical future needs rather than demonstrated current needs	"Is this component used by 3+ applications today, or am I speculating?"
Ignoring developer adoption	Designing components that are visually correct but impractical to implement	"Has a developer built with this component and confirmed it works in production?"
Accessibility as afterthought	Adding accessibility fixes after development rather than designing for accessibility from the start	"Did I specify keyboard behavior, ARIA roles, and screen reader behavior in the design spec?"
Research hoarding	Conducting research but not documenting it for reuse by other teams	"Can another Designer find and use my research findings without asking me?"
Coalition design ignorance	Designing for US users and assuming it works for coalition partners	"Have I validated terminology, date formats, and color conventions with non-US users?"
DDIL as edge case	Treating disconnected operation as a rare failure instead of a primary operating condition	"Does this application work offline? What does 'work' mean at each DDIL tier?"
Design system abandonment	Building a design system and not maintaining it — letting components drift and documentation decay	"When was the last design system release? Are all components still documented and tested?"

6-3. The Adoption Problem

The most common SL 5N failure: building a design system that no one uses. A design system that is technically excellent but ignored by application teams has zero value. Adoption requires:

1. **The system must be easier than the alternative.** If using the design system is harder than building from scratch, teams will build from scratch. Measure time-to-first-screen with and without the design system.
2. **Documentation must be complete and current.** Undocumented components are invisible components.
3. **The system must solve real problems.** Start with the components teams actually need, not the components the Designer wants to build.
4. **Developers must be involved in design system development.** Components that look correct in a design tool but break in production will be abandoned.

6-4. The Classification Complacency Risk

As Designers become comfortable working in classified environments, they may unconsciously reduce the visual prominence of classification indicators — treating them as "visual noise" rather than security controls. This is a professional hazard at the SL 5N level. Classification indicators must remain prominent, unambiguous, and visually distinct regardless of how many times the Designer has seen them. The standard is set for the least experienced user under the most stressful conditions.

SECTION 7 — THE DESIGN-DEVELOPMENT PARTNERSHIP

7-1. Why This Matters at SL 5N

BLUF: A design system that designers love but developers cannot implement is a failure. The SL 5N Designer must build for both audiences simultaneously.

At SL 4N, the Designer hands off a design spec and a developer implements it. At SL 5N, the Designer is building the shared infrastructure that both designers and developers consume. The component library is simultaneously a design artifact and a code artifact. Decisions made in the design layer — token names, component APIs, state management patterns — directly constrain what developers can build.

7-2. Coordination Points with SL 5L

Design Decision	Developer Impact	Coordination Requirement
Design token naming convention	Token names become CSS variable names and code constants	Agree on naming convention before building the token library
Component state management	Determines how components respond to data changes	Co-design state management patterns with SWE team
Responsive breakpoints	Determines layout reflow logic in code	Align breakpoints with the grid system used in production
Animation and transition specs	Performance impact on low-bandwidth/low-power devices	Validate animation specs against DDIL performance budgets
Accessibility requirements	Implementation effort for ARIA patterns	Define ARIA patterns jointly; review implementation for correctness

SECTION 8 — QUICK REFERENCE — KEY CONCEPTS

Concept	BLUF
Design system as product	Has its own backlog, roadmap, versioning, and release cycle
DDIL-first design	Design for disconnected; connectivity is a bonus
Data freshness as design variable	Every data element needs a visible age; stale data without a staleness indicator is dangerous
Classification explicitness	Classification boundaries must be unambiguous at all times; never optimize for speed across boundaries
Coalition lowest-common-denominator	Design for the least English-proficient user; no idioms, no unexpanded abbreviations
DesignOps	Processes and tools that let design quality scale; without it, quality degrades with portfolio size
Research repository	Document and tag every research finding; prevent duplicate studies; survive rotations
Accessibility is not optional	Legal requirement, operational necessity, and quality indicator; design for it from the start
Adoption over elegance	A design system nobody uses has zero value; optimize for adoption, then refine

Concept	BLUF
Developer partnership	Components must work in both design tools and production code; co-design with SL 5L

APPENDIX A — CONCEPTS GUIDE SELF-ASSESSMENT

Before proceeding to SL 5N task procedures, confirm you can answer the following from memory:

1. What distinguishes SL 5N design work from SL 4N, and why is continuing to design individual applications an underperformance at SL 5N level?
2. What are the four DDIL design tiers, and what design response does each require?
3. What three elements must accompany every data element displayed under DDIL conditions?
4. Why must classification boundary transitions require deliberate user action, and what design error does optimizing for speed across boundaries introduce?
5. Name three DesignOps functions and explain what happens to the portfolio without each.
6. What percentage of accessibility issues can automated tools catch, and what must cover the remainder?
7. What is the most common SL 5N failure mode, and what four conditions must a design system meet to achieve adoption?

APPENDIX B — CROSS-REFERENCE TO SL 5N CHAPTERS

Concepts Guide Section	Corresponding TM-50N Chapter
Section 1 (Application Design to Design Systems)	Chapter 1, Chapter 2
Section 2 (DDIL Design Thinking)	Chapter 3
Section 3 (Cross-Domain and Coalition)	Chapter 4
Section 4 (DesignOps)	Chapter 5
Section 5 (Accessibility at Enterprise Scale)	Chapter 6
Section 6 (Advanced Failure Modes)	All chapters — see Safety Summary
Section 7 (Design-Development Partnership)	Chapter 2, Chapter 5

APPENDIX C — PEER SL 5 CROSS-REFERENCES AND WFF INTEGRATION

Peer SL 5 Publications. Advanced UI/UX Designers should coordinate with practitioners in these companion advanced-track publications.

Publication	Track	Coordination Point
SL 5G	Advanced ORSA	Data visualization design for analytical products; dashboard UX for ORSA outputs
SL 5H	Advanced AI Engineer	UI/UX for AI-driven applications; presenting model outputs and uncertainty to users
SL 5M	Advanced ML Engineer	Visualization of ML model performance; feature importance displays
SL 5J	Advanced Program Manager	Portfolio-level product strategy; design system roadmap prioritization
SL 5K	Advanced Knowledge Manager	Knowledge portal UX; search interface design; taxonomy visualization
SL 5L	Advanced Software Engineer	Design system implementation architecture; shared component libraries; token-to-code pipeline
SL 5O	Advanced Platform Engineer	Platform portal design; performance budgets; deployment constraints affecting design patterns

WFF Operational Consumer Note. Design systems and UX patterns built by SL 5N designers are consumed by the six Warfighting Function (WFF) tracks: Intelligence (SL 4A), Fires (SL 4B), Movement and Maneuver (SL 4C), Sustainment (SL 4D), Protection (SL 4E), and Mission Command (SL 4F). These practitioners are the operational users of every dashboard, COP layer, and decision support product in the portfolio. The design questions addressed in this Concepts Guide — DDIL resilience, classification explicitness, data freshness indicators, accessibility — must be answered in terms of what a WFF staff section needs to make a decision under operational conditions. A G2 intelligence dashboard has different layout priorities than a G4 sustainment status board; both must be accessible, DDIL-resilient, and unambiguous about data freshness.

CONCEPTS GUIDE — SL 5N COMPANION // ADVANCED UI/UX DESIGNER HEADQUARTERS, UNITED STATES ARMY EUROPE AND AFRICA // WIESBADEN, GERMANY // 2026