# UI Implementation Considerations

Dynamic Entity Hierarchy with Entity Kit Model

## Executive Summary

Moving from a rigid, level-based system to a flexible Entity Kit model requires significant UI/UX changes. The current implementation assumes specific levels (e.g., Parent) for onboarding, products, and coverage. The Entity Kit approach enables these actions at **any hierarchy level**, but this flexibility introduces new challenges around user guidance, workflow initiation, and data integrity.

### Current State vs. Entity Kit Approach

| Aspect | Current (Rigid) | Entity Kit (Flexible) |
|---|---|---|
| Onboarding | Auto-triggered at Parent level | Manual "Initiate Onboarding" at any level |
| Product Assignment | Fixed to specific level | Attachable at any hierarchy level |
| Coverage | On entity only | On entity OR on specific product |
| Workflow | One-size-fits-all | Product-specific guidance |
| User Decisions | Implicit assumptions | Explicit user choices |

# Key UI/UX Considerations & Gotchas

## 1. Onboarding Initiation Problem

***GOTCHA: If onboarding isn't automatic, how does a user know WHEN and WHERE to trigger it?***

**Considerations:**

- Need an explicit 'Initiate Onboarding' action (button/menu)
- User must understand their product desk's typical pattern
- Risk: Users create entities but forget to onboard → incomplete data

**Possible UI Solutions:**

> **Option A:** "Initiate Onboarding" button on any Entity Kit
> Every entity shows available actions regardless of level.
>
> **Option B:** Product attachment triggers onboarding prompt
> "You're adding Product 2 to this entity. Does this entity require KYC/onboarding? [Yes] [No, inherited]"
>
> **Option C:** Guided workflow based on product desk rules
> "Product 2 is typically onboarded at Child level. Would you like to start onboarding for this entity?"

## 2. The "Where Does This Belong?" Problem

***GOTCHA: Users may not know which level is appropriate for their action.***

**Considerations:**

- Different product desks have different conventions
- New users won't know the 'typical' patterns
- Mistakes could mean duplicate KYC or missed compliance

**Possible UI Solutions:**

> **Option A:** Product-specific guidance
> When adding a product, show: "Product 1 is typically attached at Parent level. You're attaching to a Child. Continue anyway?"
>
> **Option B:** Visual indicators of "expected" vs "actual"
> Show typical level with subtle highlight; flag entities that deviate from pattern (not as errors, just as 'unusual').
>
> **Option C:** Templates / Presets
> "Set up for Product 1 workflow" → pre-configures expected levels; "Set up for Product 2 workflow" → different configuration.

## 3. The Inheritance vs. Explicit Problem

***GOTCHA: If a Parent has KYC, does the Child inherit it? Or does each level need its own?***

**Considerations:**

- Some documents apply to the whole hierarchy (legal entity docs)
- Some documents are specific to a level (fund-specific agreements)
- Coverage might inherit down or be overridden

**Possible UI Solutions:**

> **Option A:** Explicit inheritance indicators
> Show "Inherited from Parent ↑" vs "Local to this entity" for each field.
>
> **Option B:** "Inherited" vs "Local" toggle
> Each field shows: [Inherit from Parent] [Set Locally] toggle.
>
> **Option C:** Inheritance rules per field type
> Documents: Always local | Coverage: Configurable | Products: Never inherited (explicit only).

## 4. The "Empty Entity Kit" Problem

***GOTCHA: What does an entity look like before anything is configured?***

**Considerations:**

- Entity exists in hierarchy but has no products, no coverage, no docs
- Is this valid? Or a 'draft' state?
- How do you find 'incomplete' entities?

**Possible UI Solutions:**

> **Option A:** Status indicators
> Show [Draft], [Active], [Pending Setup] badges. Include "This entity needs setup" warning with [Complete Setup Wizard] button.
>
> **Option B:** Required vs Optional indicators
> Show which fields are REQUIRED for this entity to be 'active' vs which are optional enhancements.
>
> **Option C:** Dashboard / Work Queue
> "Entities Pending Setup" list showing incomplete Entity Kits for follow-up.

## 5. The "Who Can Do What" Problem (Permissions)

***GOTCHA: If any level can have products/coverage, who has permission to configure each level?***

**Considerations:**

- Product Desk A might only manage Parent-level entities
- Product Desk B might only manage Child-level entities
- But the system allows both to do everything

**Possible UI Solutions:**

> **Option A:** Role-based visibility
> Product Desk A users only SEE the 'Add Product' button at Parent level. System allows both, but UI restricts by role.
>
> **Option B:** Approval workflows
> "You're adding Product 1 at Child level (unusual). This requires approval from [Admin]."
>
> **Option C:** Soft guidance, hard audit
> Anyone CAN do anything, but unusual actions are logged/flagged. Compliance reviews 'exceptions' periodically.

## 6. The "Multiple Products, Multiple Workflows" Problem

***GOTCHA: If an entity has Product 1 AND Product 2, they may have different onboarding requirements.***

**Considerations:**

- Product 1 might need documents A, B, C

- Product 2 might need documents B, D, E
- Do you show combined requirements? Separate tracks?

**Possible UI Solutions:**

**Option A:** Combined checklist with product tags
Show all required docs with tags indicating which product requires each: "Doc B (Product 1, Product 2)".

**Option B:** Tabbed by product
[Product 1 Onboarding] [Product 2 Onboarding] tabs, each showing its own workflow independently.

**Option C:** Unified status with drill-down
"Onboarding: 60% complete" — click to see breakdown by product.

## 7. The "Transaction Code Mapping" Problem

***GOTCHA: Transactions happen against a "Code" — where does that Code live now?***

**Considerations:**

- Currently Code is on the Parent (entity level)
- With Entity Kit, Code could be on any level OR on the Product attachment
- Need to maintain backward compatibility with existing transaction systems

**Possible UI Solutions:**

> **Option A:** Code lives on Entity, Products reference it
> Entity has the Code; Products attached use that Code. Simple, but doesn't support different codes per product.
>
> **Option B:** Code lives on Product Attachment
> Each Product attachment has its own Code. Entity-level Code is optional/legacy.
>
> **Option C:** Code mapping layer
> Transactions come in with a Code. System maps Code → Entity + Product based on configuration. UI shows "Transaction Routing Rules".

# Summary: What Changes in UI

| Current UI | Entity Kit UI |
|---|---|
| Create Parent → Auto-onboarding | Create Entity → Manual "Initiate Onboarding" |
| Product assigned at fixed level | Product can be attached at any level |
| Coverage on entity | Coverage on entity OR on specific product |
| One workflow fits all | Product-specific workflow guidance |
| Implicit assumptions | Explicit user choices |
| Simple but rigid | Flexible but requires more user decisions |

# Recommendation: Phased Approach

## Phase 1: Add flexibility without removing guardrails

- Keep current defaults (Parent = onboarding level)
- Add 'override' capability for exceptions
- Log all non-standard configurations for audit

## Phase 2: Product-aware guidance

- Implement product desk 'profiles' that suggest typical patterns
- Warn (don't block) when users deviate from expected patterns
- Add visual indicators for 'typical' vs 'exception' configurations

## Phase 3: Full flexibility

- Remove hardcoded assumptions
- All behavior driven by product desk configuration
- Robust audit trail for compliance
- Self-service configuration for product desk admins

# Key Decisions Required

| # | Decision | Options | Impact |
|---|---|---|---|
| 1 | How to initiate onboarding? | Auto / Manual / Product-triggered | Workflow design |
| 2 | Where does Coverage live? | Entity / Product / Both | Data model |
| 3 | How to handle inheritance? | Explicit / Implicit / Configurable | UI complexity |
| 4 | Where does transaction Code live? | Entity / Product / Mapping layer | Integration |
| 5 | How to guide users? | Hard rules / Soft guidance / Templates | User experience |
| 6 | How to handle permissions? | Role-based / Approval workflow / Audit | Security model |

# Next Steps

1. Review this document with product desk stakeholders
1. Prioritize which gotchas are most critical for your use cases
1. Decide on phased approach timeline
1. Create UI wireframes/mockups for selected solutions

1. Define data model changes required
1. Plan backward compatibility strategy