Làm thế nào để đạt được mô hình tư duy chung

Việc đạt được mô hình tư duy chung(**shared mental model)** là **một trong những mục tiêu quan trọng nhất trong DDD** — và cũng là một trong những **khó khăn lớn nhất** nếu không làm đúng cách.

Những rào cản lớn để đạt được

Rào cản	Tác động
Thiếu giao tiếp giữa dev và domain expert	Không hiểu đúng nghiệp vụ
Ngôn ngữ không thống nhất	Dễ gây nhầm lẫn khi thảo luận hoặc code
Developer chỉ tập trung vào kỹ thuật	Bỏ qua logic nghiệp vụ, hiểu sai bản chất
Domain expert không quen mô hình hóa	Không diễn đạt đúng ý hoặc bỏ sót ý quan trọng
Áp lực deadline → code trước, hiểu sau	Dẫn đến phần mềm xa rời thực tế
Không ghi chú/tài liệu hoá rõ ràng	Không lưu giữ tri thức đã chia sẻ

Agile – Giải pháp thay thế cho mô hình truyền thống

Tuyên ngôn Agile đề xuất chuyển dịch:

- Từ **phát triển dựa trên tài liệu s**ang **cách tiếp cận tập trung vào hợp tác**
- Thoát khỏi tư duy **tổ chức phân mảnh (siloed organization)**
- Xây dựng **niềm tin** và **chia sẻ kinh nghiệm**

2 giá trị cốt lõi của Agile giải quyết thách thức giao tiếp

1. **Ưu tiên tương tác giữa con người** hơn quy trình hoặc công cụ
2. **Hợp tác với khách hàng** thay vì đàm phán hợp đồng cứng nhắc
   → Cả hai nhấn mạnh **làm việc chặt chẽ với khách hàng** để xây dựng **mô hình tư duy chung**

Cách Agile xây dựng mô hình tư duy chung

- **Giao tiếp mở và thường xuyên** giữa kỹ sư và khách hàng để giải quyết bất đồng
- **Phương pháp lặp (iterative)**: Như Scrum hay XP (Extreme Programming) tập trung vào:
  - Phản hồi liên tục từ khách hàng/đội ngũ
  - Kế hoạch linh hoạt thay vì chi tiết cứng nhắc
- **Học tập tương tác**: Hiểu sâu vấn đề của người dùng thông qua:
  - Thảo luận trực tiếp
  - Mockup/prototype đơn giản
    → Kích thích trao đổi và điều chỉnh sớm

Vai trò của Prototyping

- **Prototype song song**: Thử nghiệm nhiều giải pháp cùng lúc để tìm phương án tối ưu
- **Tiến hóa từ mockup thô → sản phẩm tinh chỉnh**:
  - Mỗi vòng lặp làm sâu sắc thêm hiểu biết về vấn đề
  - Kế hoạch luôn giữ tính linh hoạt

Cơ chế phản hồi trong Agile

1. **Lập trình cặp (Pair Programming)/Mob Programming**:
   - Kết hợp góc nhìn đa dạng → Hiểu biết tập thể
   - Khuyến khích **khách hàng tham gia trực tiếp** để giảm hiểu lầm
2. **Phát triển hướng kiểm thử (TDD)**:
   - Viết test trước code → Đảm bảo phần mềm đáp ứng nghiệp vụ
   - Test nên viết từ góc độ **yêu cầu kinh doanh**, không chỉ kỹ thuật
3. **Stand-up Meeting**:
   - Giải quyết bất đồng ngay lập tức
   - Duy trì sự đồng bộ trong team
4. **Planning Meeting**:
   - Sự tham gia của khách hàng → Xác nhận mục tiêu chung

Kết quả đạt được

- Phần mềm **linh hoạt**, **đúng nhu cầu** nhờ:
  - Giao tiếp liên tục
  - Phản hồi theo vòng lặp
- **Tài liệu sống động** (living documentation) thay vì tĩnh

Kết nối giữa Agile và Domain-Driven Design (DDD)

- **DDD bổ sung cho Agile** bằng:
  - **Ubiquitous Language**: Ngôn ngữ chung giữa kỹ thuật & nghiệp vụ
  - **Bounded Context**: Xác định phạm vi rõ ràng
  - **Event Storming**: Công cụ trực quan hóa bài toán nghiệp vụ

Tác giả: Đỗ Ngọc Tú

Công Ty Phần Mềm VHTSoft

Phiên bản #1
Được tạo 22 tháng 4 2025 01:53:14 bởi Đỗ Ngọc Tú
Được cập nhật 22 tháng 4 2025 01:59:01 bởi Đỗ Ngọc Tú