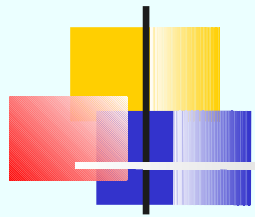


Công nghệ phần mềm **Nhóm (team)** **phát triển phần mềm**

Giảng viên: TS. Nguyễn Mạnh Hùng
Học viện Công nghệ Bưu chính Viễn thông (PTIT)



Tổ chức nhóm PTPM

Trên lí thuyết thì:

- Nếu một sản phẩm phần mềm phải giao trong 3 tháng, nhưng đòi hỏi khối lượng công việc là 12 tháng/người
- → Dùng 4 người phát triển phần mềm đó thì có đúng hạn và chất lượng không?



Chia sẻ công việc (1)

- Một nông dân cày đám ruộng hết 10 ngày
→ nếu có 10 nông dân cày đồng thời thì chỉ hết có 1 ngày
- Một phụ nữ trong 9 tháng thì sinh được 1 baby
→ 9 phụ nữ có thể sinh 1 baby trong 1 tháng?



Chia sẻ công việc (2)

- Không giống việc sinh baby, phát triển phần mềm là một dạng công việc có thể chia sẻ được
- Cũng không giống cày ruộng, PTPM cần đến các kĩ năng hợp tác trong nhóm hiệu quả



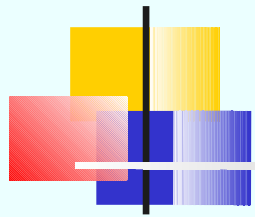
Tổ chức nhóm code (1)

Xét ví dụ:

- A và B phải code hai modul M1 và M2

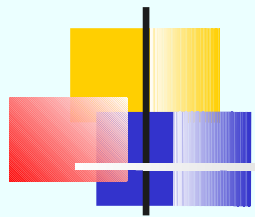
Các lỗi sau có thể xảy ra:

- A và B cùng code M1, nghĩ rằng người còn lại code M2
- A code M1, B code M2. M1 gọi M2 truyền 4 tham số, nhưng M2 nhận 5 tham số
- Hai bên đều có 4 tham số, nhưng thứ tự và kiểu tham số bên gọi khác bên định nghĩa



Tổ chức nhóm code (2)

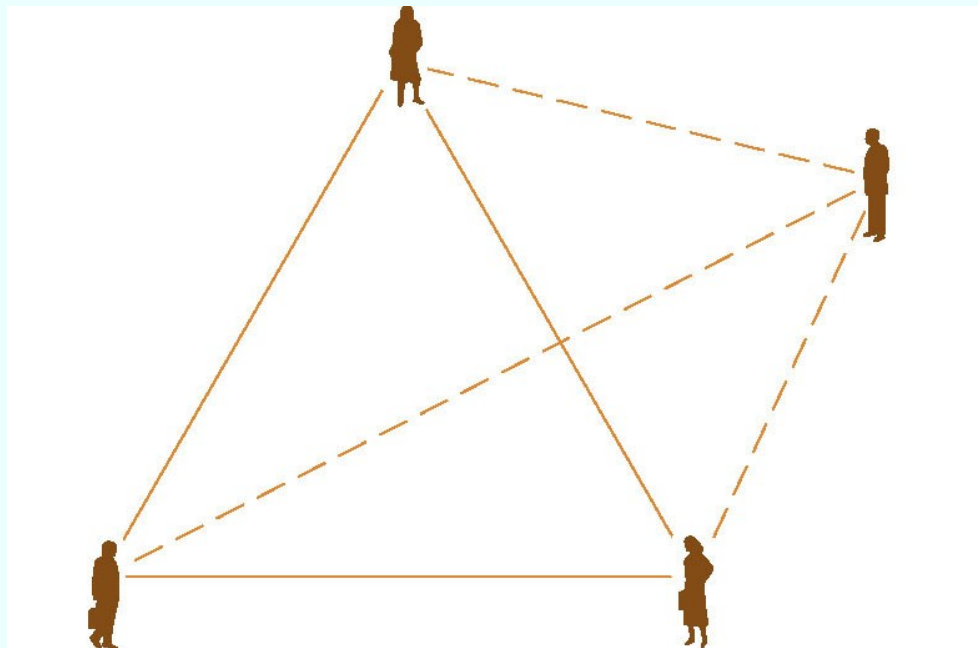
- Không phải vấn đề về năng lực kĩ thuật
→ mà là vấn đề quản lí con người và công việc!



Vấn đề giao tiếp (1)

Xét ví dụ:

- Đội phát triển có 3 người → có 3 kênh giao tiếp.
Nhưng dự án sắp đến hạn mà còn quá nhiều việc
- Giải pháp trực quan: tuyển thêm 1 người
→ cần 6 kênh giao tiếp!





Vấn đề giao tiếp (2)

3 người cũ sẽ phải diễn giải cho người mới:

- Các việc đã hoàn thành
- Các việc chưa hoàn thành
- Cách hoàn thiện các việc còn dang dở

Luật Brooks:

- Khi đưa thêm người mới vào dự án đang nguy cơ bị trễ, thì không giải quyết được vấn đề trễ, thậm chí còn làm dự án bị trễ thêm!



Tổ chức nhóm PTPM

Thông thường:

- Nhóm PTPM làm việc với nhau trong suốt tiến trình PTPM, nhưng quan trọng nhất là giai đoạn code

→ người ta quan tâm đến việc tổ chức nhóm code

Có hai loại nhóm code:

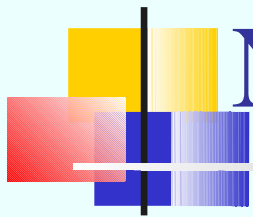
- Nhóm code bình đẳng (dân chủ)
- Nhóm code có sếp



Nhóm code bình đẳng (1)

Dựa trên các nguyên tắc:

- Các thành viên bình đẳng về chức vụ
- Mỗi người tự do thiết kế, code và test modul của mình
- Việc có lỗi được coi là việc bình thường
- Cả đội sẽ xây dựng một tính năng hay cả sản phẩm, và sản phẩm là của cả đội



Nhóm code bình đẳng (2)

Thuận lợi:

- Các thành viên nắm chắc phần code của mình
- Khả năng code mạnh, nhất là giải quyết các dự án khó

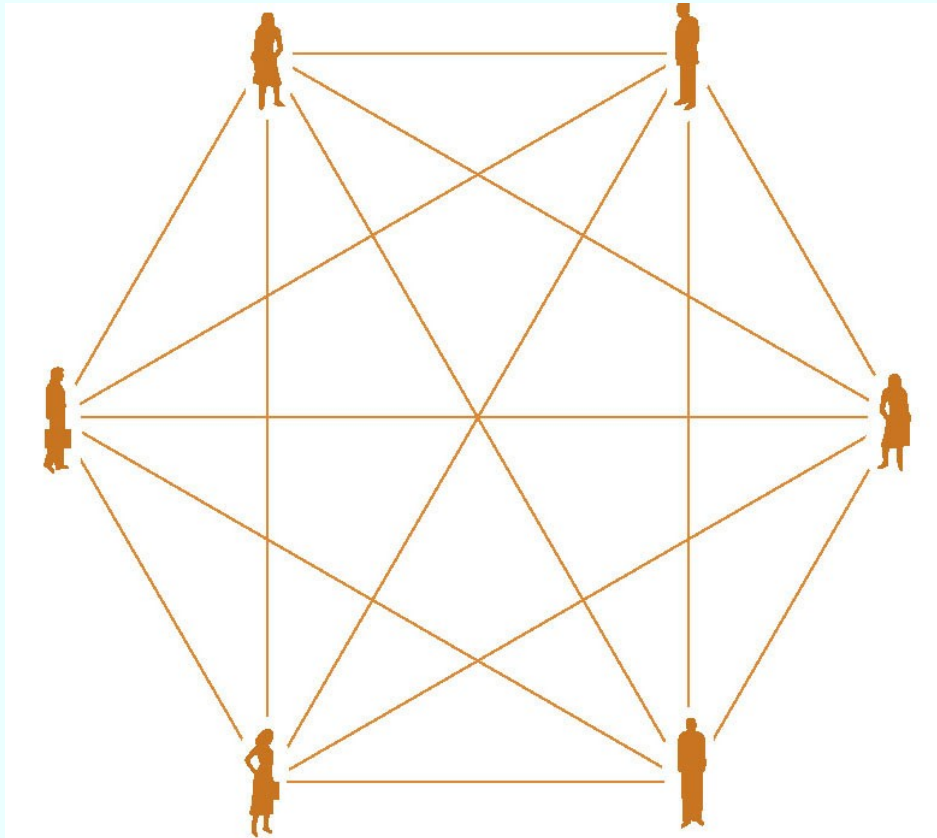
Khó khăn:

- Việc tự test code của mình thường không hiệu quả
 - Khó khăn về mặt quản lí
- Đội phải được phát triển một cách tự nhiên

Nhóm code có sếp – kiểu cũ (1)

Xem xét nhóm có 6 người:

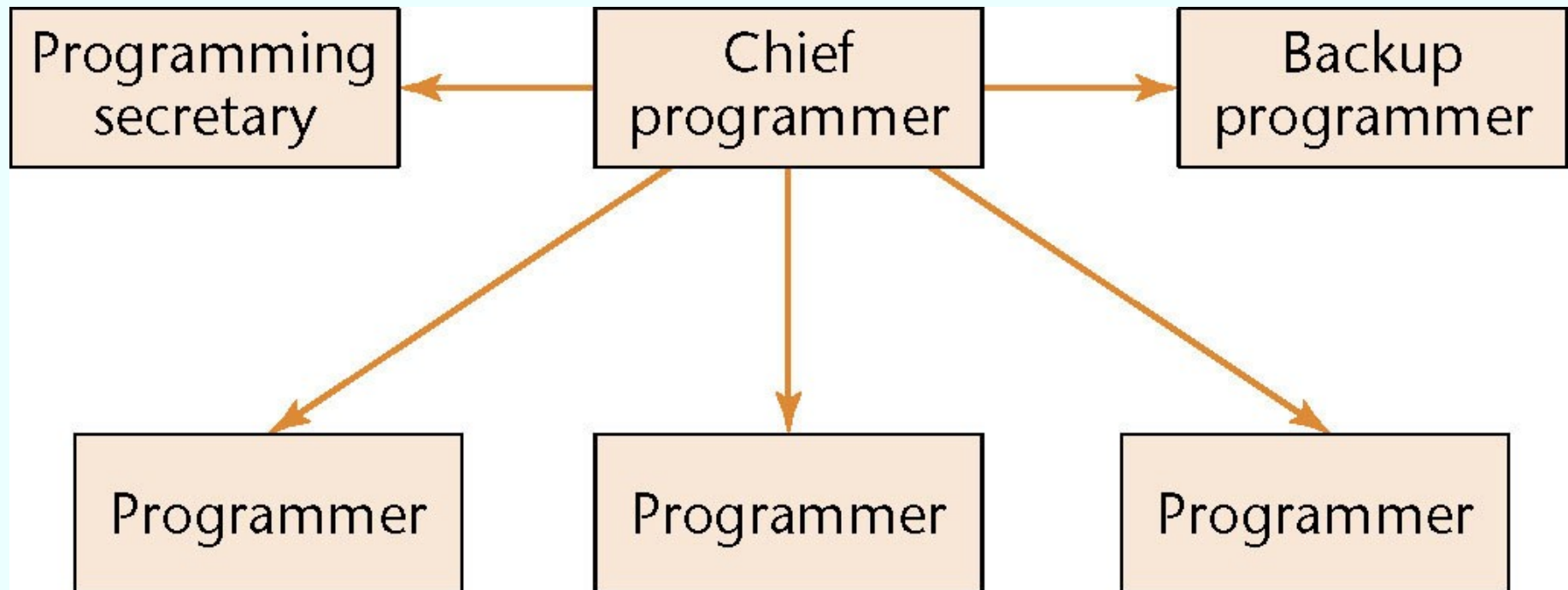
- Có 15 cặp giao tiếp
- Tổng các nhóm con có 2, 3, 4, 5, 6 người là 57
- Nhóm này không thể giải quyết 1 việc cần 6 tháng/ người trong 1 tháng!



Nhóm code có sếp – kiểu cũ (2)

Xem xét giải pháp:

- Có 6 thành viên, nhưng chỉ còn 5 cặp giao tiếp!
- Dựa trên ý tưởng nhóm bác sĩ của 1 ca phẫu thuật





Nhóm code có sếp – kiểu cũ (3)

Sếp của nhóm code:

- Có kĩ năng cao trong quản lí và code
- Thực hiện phần thiết kế kiến trúc
- Phân công công việc code cho các thành viên
- Code các phần chính và khó nhất
- Tạo các giao diện để tích hợp các modul
- Xem lại code của tất cả các thành viên
- Chịu trách nhiệm về từng dòng code của nhóm



Nhóm code có sếp – kiểu cũ (4)

Sếp dự bị của nhóm code:

- Dự bị cho sếp của nhóm
- Có kĩ năng tương đương sếp trong quản lí và code
- Hiểu rõ dự án tương đương sếp
- Lập kế hoạch test hộp đen (black-box) và các công việc độc lập với tiến trình thiết kế



Nhóm code có sếp – kiểu cũ (5)

Thư kí lập trình của nhóm code:

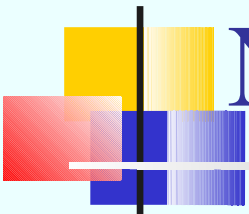
- Có kĩ năng cao, trả lương cao, và là thành viên chủ chốt của nhóm
- Chịu trách nhiệm về tài liệu cho toàn bộ dự án:
 - Liệt kê danh sách mã nguồn
 - Ngôn ngữ điều khiển công việc (JCL)
 - Dữ liệu test
 - Biên dịch code, kiểm tra code convention
 - Chạy các test case



Nhóm code có sếp – kiểu cũ (6)

Lập trình viên:

- Chỉ làm việc duy nhất là lập trình
- Các việc khác liên quan đã có thư kí lập trình lo!



Nhóm code có sếp – kiểu cũ (7)

Khó khăn:

- Sếp, dự bị đều phải có đồng thời kĩ năng cao trong cả quản lí và code. Nhưng thường người quản lí giỏi thì code kém và ngược lại
- Sếp dự bị phải có kĩ năng tương đương sếp, nhưng phải làm dự bị cho sếp và trả lương thấp hơn → khó ai chấp nhận!
- Thư kí không làm gì ngoài việc làm tài liệu cả ngày → lập trình viên thường ghét việc làm tài liệu!



Mô hình nhóm kết hợp (1)

Mục đích:

- Kết hợp ưu điểm của cả hai mô hình:
 - Nhóm bình đẳng: tinh thần phát hiện và sửa lỗi cao
 - Nhóm có sếp: quản lí và giao tiếp tốt

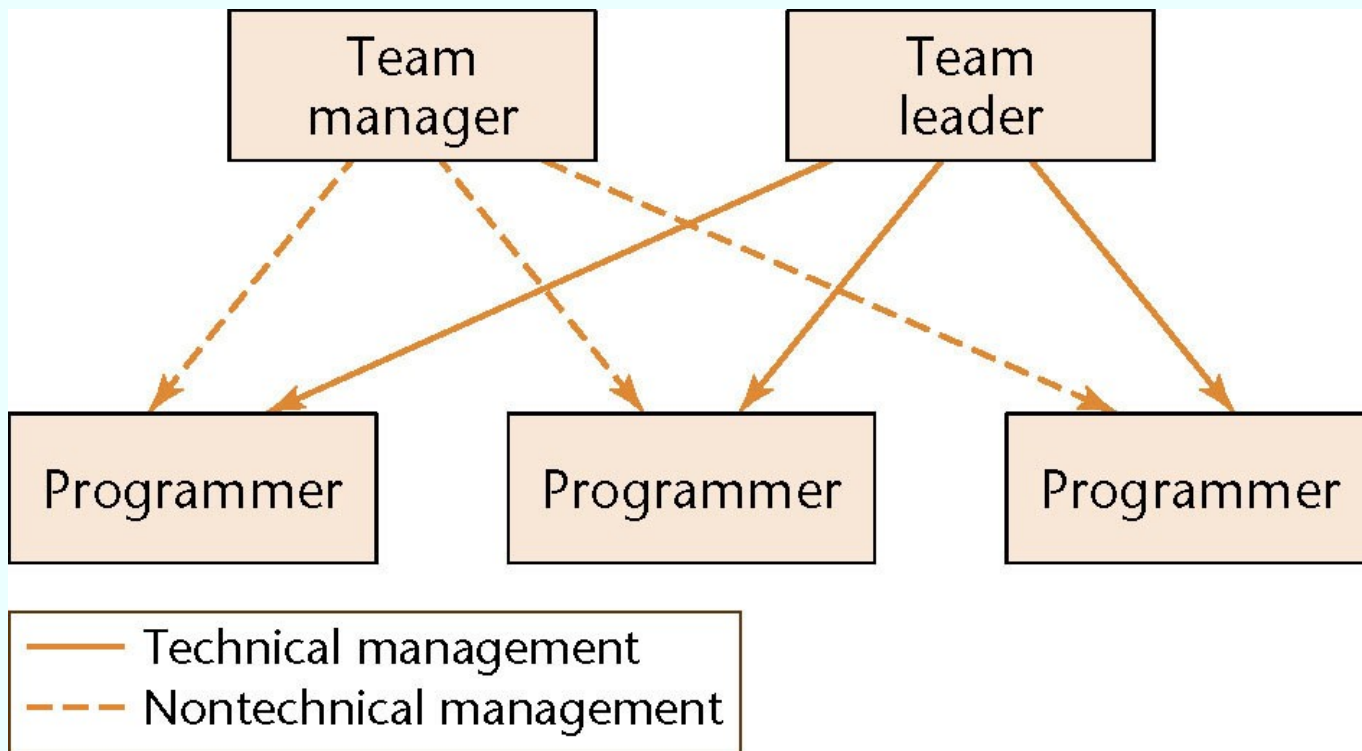
Thực tế, trong mô hình CPT:

- Sếp chịu trách nhiệm về từng dòng code nên phải review toàn bộ code
 - Sếp cũng chịu trách nhiệm quản lí nên có thể không cần review code
- Giảm bớt trách nhiệm của sếp!

Mô hình nhóm kết hợp (2)

Giải pháp:

- Sếp chỉ quản lí các vấn đề phi kĩ thuật
- Tạo ra team leader để quản lí kĩ thuật





Mô hình nhóm kết hợp (3)

Hoạt động:

- Sếp chỉ quản lí các vấn đề phi kĩ thuật : thu nhập, bình đẳng, năng lực của các thành viên
- Team leader chỉ quản lí kĩ thuật: review toàn bộ code và hỗ trợ kĩ thuật cho các thành viên
- Sếp không review code nhưng khi họp có thể tham gia để hỗ trợ kĩ thuật cho các thành viên



Mô hình nhóm kết hợp (4)

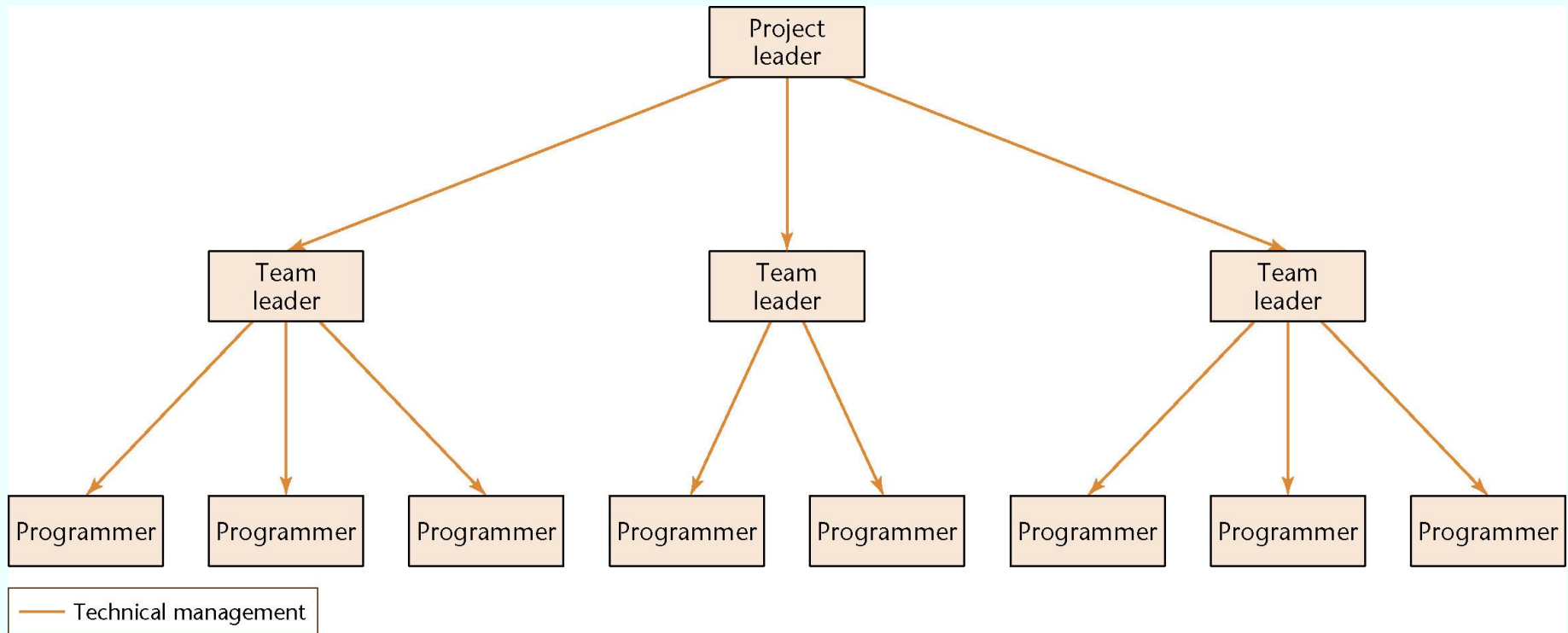
Tính khả thi:

- Tìm một team leader dễ dàng hơn nhiều một sếp
- Các thành viên chỉ bị quản lí bởi 1 sếp duy nhất
- Sếp chỉ cần kĩ năng cao về quản lí, team leader chỉ cần kĩ năng cao về code → dễ tìm hơn

Mô hình nhóm kết hợp (5)

Với dự án lớn:

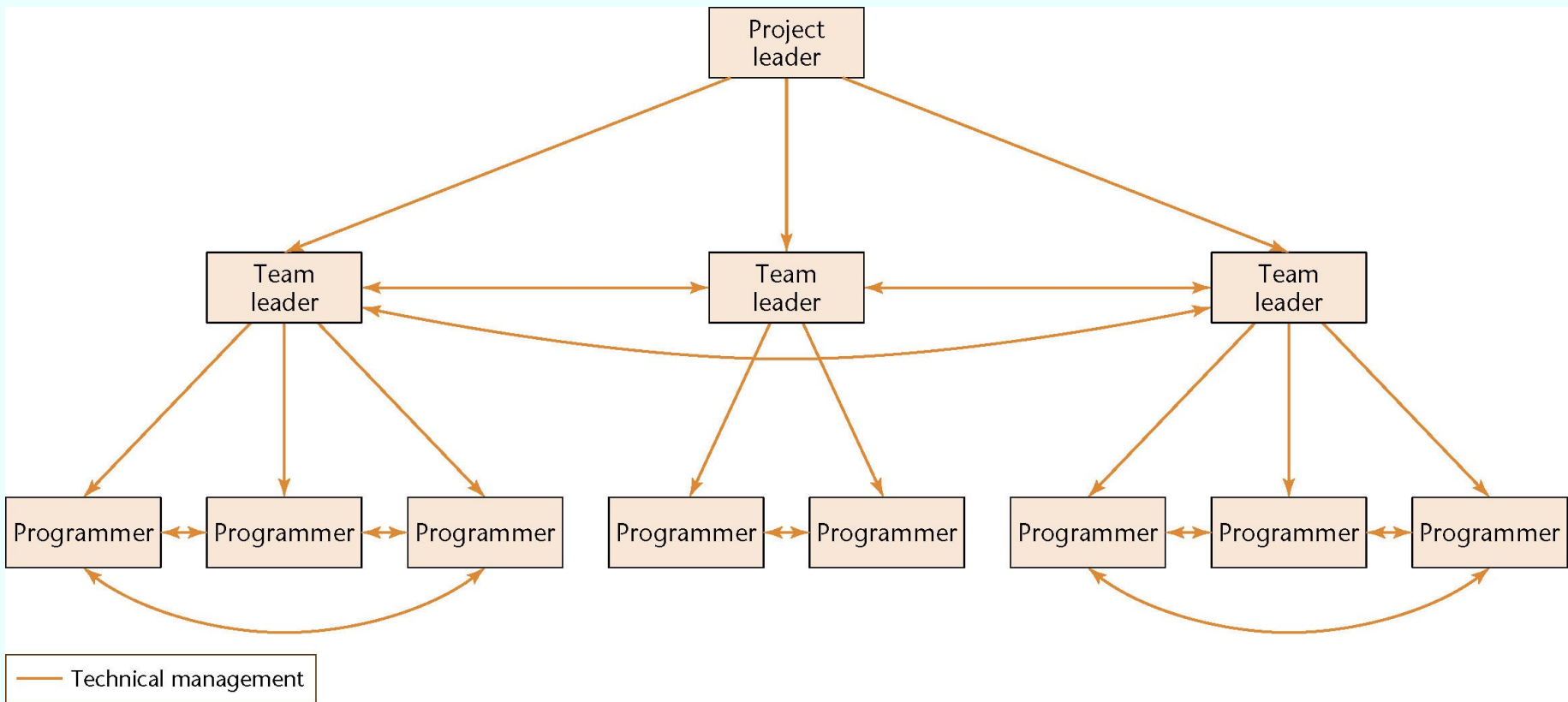
- Thêm một tầng quản lý kỹ thuật

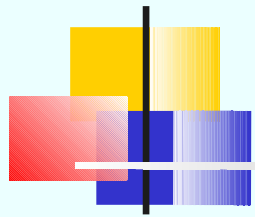


Mô hình nhóm kết hợp (6)

Vấn đề ra quyết định:

- Dùng phương pháp nhóm bình đẳng





Nhóm code cho mô hình XP

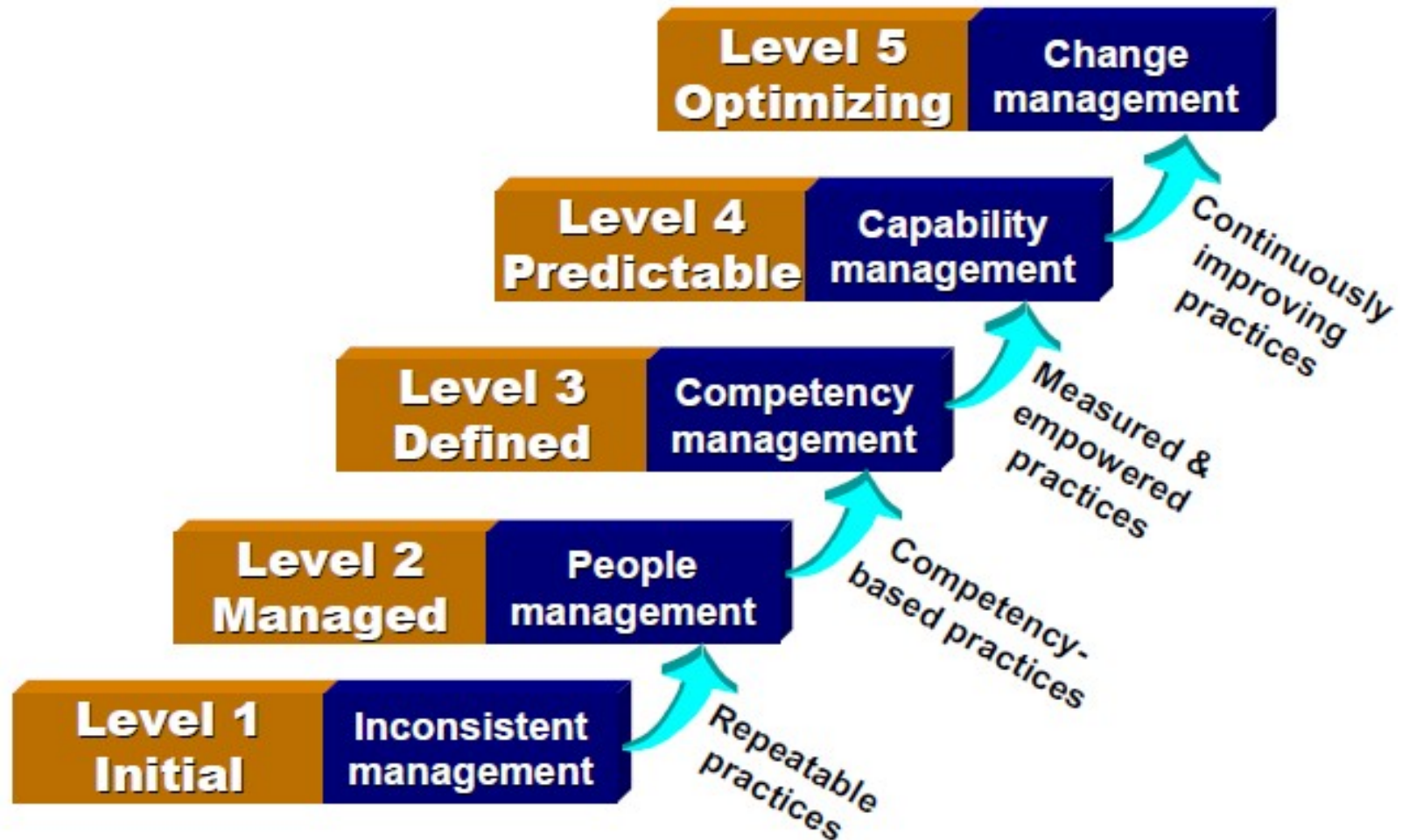
Mô hình:

- Lập trình theo cặp, mỗi cặp chung máy
- Test chéo: người này test code của người kia

Ưu điểm:

- Khi một thành viên rời nhóm, thì thành viên mới dễ dàng gia nhập vì có thể học với người cùng cặp
- Phát huy được ưu điểm của lập trình bình đẳng

People - CMM





Questions?
