GPS - Gestão de Projetos de Software

Aula 7: Comunicação e Stakeholders

Prof. Aléssio Miranda Júnior
alessio@cefetmg.br

CEFET-MG - Campus Timoteo
Dep. Engenharia de Computação

Outubro de 2026

Sumário

Objetivos da Aula

- Definir o conceito corporativo de **Stakeholder** (Parte Interessada).
- Mapear e classificar stakeholders na **Matriz de Poder vs. Interesse**.
- Calcular a explosão de complexidade usando a **Fórmula de Canais de Comunicação**.
- Diferenciar taticamente as ferramentas **Síncronas** e **Assíncronas**.
- Aprender as **5 Abordagens de Conflitos** do PMBOK.
- Dominar a negociação técnica do "Efeito Caixa Preta" na Era da IA.
- Organizar os critérios definitivos do **Check 02** no GitLab.

O Mentor Adverte

O Verdadeiro Bug de Produção

"Garoto, projetos de software falham raramente por conta de ponteiros nulos, falha de compilador ou falta de memória RAM; eles falham de forma miserável porque as pessoas não se entendem."

Neste sétimo capítulo, encerramos todo o ciclo da **Gestão Tradicional Preditiva** lidando com o componente mais instável, caro e volátil do sistema: os humanos. Falar a coisa certa, para a pessoa certa, no canal certo, é o que separa os líderes dos amadores.

O que é um Stakeholder?

O termo **Stakeholder** (Parte Interessada) não se resume ao clichê do "cliente engravatado que paga pelo projeto".

Definição Ampla do PMI

Refere-se a qualquer indivíduo, grupo ou organização que pode afetar, ser afetado ou *se perceber* afetado pelas decisões e entregas do seu projeto.

No **Projeto Assistente IA**, os stakeholders são vastos:

- A sua própria equipe (desenvolvedores, arquitetos, QA).
- O Professor (Sponsor / Patrocinador).
- Os alunos da maratona (Usuários Finais).
- A coordenação do CEFET (Fornece a infraestrutura e laboratórios).

O Esgotamento do Gerente

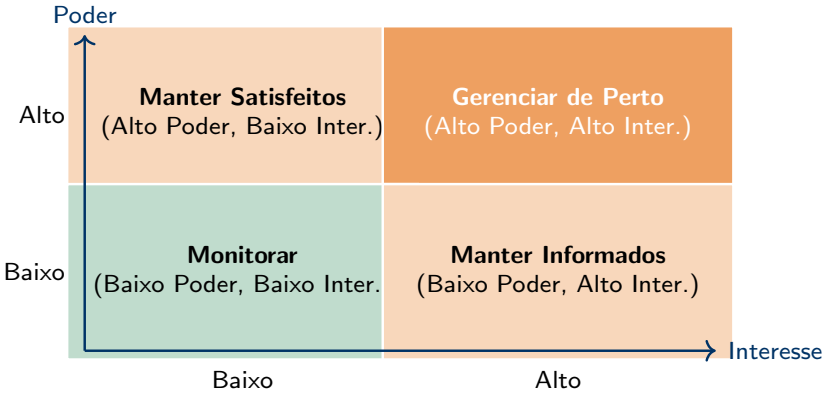
É humanamente impossível (e indesejável do ponto de vista gerencial) enviar relatórios detalhados e agradar a todos os stakeholders o tempo todo.

- Alguns precisam validar e aprovar cada tela do sistema.
- Outros só querem saber se o projeto não estourou o prazo.
- Outros só querem um tutorial de como instalar na véspera do lançamento.

Para triar esse nível de engajamento sem enlouquecer, a gerência de projetos utiliza uma ferramenta visual: a **Matriz de Poder vs. Interesse**.

Matriz de Poder vs. Interesse

Cruzamos o **Poder** de veto e decisão com o **Interesse** no dia a dia do desenvolvimento.



Quadrante 1: Gerenciar de Perto

Alto Poder + Alto Interesse

- **Quem é?** O Sponsor (Patrocinador), o Product Owner, ou o Cliente Direto (Professor da disciplina).
- **O Nível de Poder:** Eles aprovam as entregas, o escopo e o orçamento. Têm o poder de cancelar o projeto.
- **O que fazer?** Eles exigem **comunicação massiva**. Precisam de reuniões frequentes, demonstrações contínuas, validação da arquitetura (ex: validação da IA escolhida) e protótipos funcionais. Você deve dormir e acordar alinhado com eles.

Quadrante 2: Manter Satisfeitos

Alto Poder + Baixo Interesse

- **Quem é?** A Diretoria da empresa, o Governo, ou a Coordenação do Curso.
- **O Nível de Poder:** Eles fornecem os laboratórios ou as regras legais. Podem suspender o projeto por compliance, mas não se importam com as minúcias técnicas.
- **O que fazer?** Eles não querem saber se você usou Redis, RabbitMQ ou arquivo JSON. Envie **relatórios executivos concisos** (1 página). Ex: "Estamos no prazo, dentro do orçamento e sem violações". Se você lotá-los de detalhes, eles ficarão irritados.

Quadrante 3: Manter Informados

Baixo Poder + Alto Interesse

- **Quem é?** Os usuários finais (No nosso caso: Alunos Maratonistas).
- **O Nível de Poder:** Eles não ditam a arquitetura, não controlam o dinheiro e não mandam refazer o sistema. Mas o interesse deles é alto porque vão usar o software diariamente!
- **O que fazer?** Envie *Release Notes*, newsletters, e-mails avisando sobre novidades e tutoriais de como usar o novo *Chatbot*. Mantenha um canal de *Feedback* para captar os *bugs* que eles acharem.

Quadrante 4: Monitorar

Baixo Poder + Baixo Interesse

- **Quem é?** Concorrentes, público geral não impactado, outras turmas.
- **O Nível de Poder:** Baixíssimo envolvimento nas duas escalas.
- **O que fazer? Esforço mínimo**. Apenas os monitore de longe para garantir que o nível de poder ou interesse deles não mude repentinamente, ou que não criem obstáculos burocráticos imprevistos.

A Ilusão da "Comunicação Orgânica"

A armadilha clássica do desenvolvedor

"Ah, nosso time é pequeno. A gente não precisa de um plano formal ou ferramentas corporativas. A gente se fala direto no grupo do WhatsApp e resolve rápido."

Esse pensamento destrói a escalabilidade de qualquer projeto. A comunicação humana cresce de forma **exponencial**, e não linear, toda vez que novas pessoas entram na equipe de engenharia.

A Lei Matemática da Complexidade

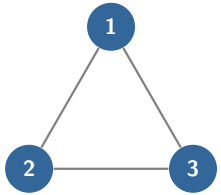
O PMBOK estabelece uma fórmula matemática rígida para calcular a quantidade de **Canais de Comunicação** (C) baseada exclusivamente no número de pessoas do projeto (N):

$$C = \frac{N(N-1)}{2}$$

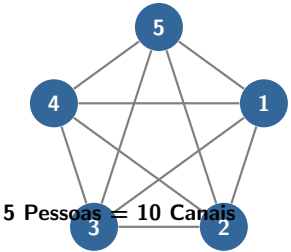
- Se o número de stakeholders apenas dobra, a complexidade (quantidade de nós e ligações) *quadruplica* instantaneamente.
- O seu cérebro de gerente não tem banda larga o suficiente para acompanhar e lembrar de 45 conversas soltas nos corredores. A informalidade rui sob o próprio peso.

A Explosão Visual de Canais

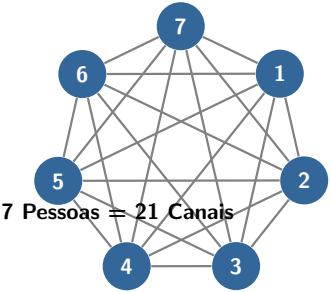
Veja como adicionar apenas poucas pessoas transforma a gestão em um pesadelo:



3 Pessoas = 3 Canais



5 Pessoas = 10 Canais



7 Pessoas = 21 Canais

Síncrono vs Assíncrono: A Batalha do Foco

O **Plano de Comunicação** serve para domar os Canais, definindo O QUÊ transmitir, para QUEM e POR ONDE, respeitando o foco da equipe técnica.

1. **Canais Síncronos (Reuniões, Calls, Meet):**
   - São caros (somam o valor-hora de todos os presentes simultaneamente).
   - Destroem impiedosamente o estado de foco (*flow*) do programador.
   - *Uso correto:* Apenas para resolução de crises graves ou brainstormings táticos.
2. **Canais Assíncronos (GitLab, Slack, Teams, E-mail):**
   - O engenheiro escolhe ler no momento apropriado do seu ciclo de trabalho.
   - Tudo fica documentado, rastreável e versionado.
   - *Uso correto:* Status diário, alertas automatizados de *deploy*, dúvidas e *Code Reviews*.

O Perigo Radioativo do WhatsApp

O Cemitério Silencioso de Decisões

O WhatsApp é excelente para avisar que o pneu do ônibus furou, mas é **completamente radioativo** para a gestão de um software corporativo.

- Mudanças críticas de requisitos feitas no zap desaparecem entre figurinhas e áudios.
- Exclui violentamente do contexto os *stakeholders* que estão offline naquele minuto.
- **A Regra de Ouro Corporativa:** Se a decisão técnica não está cravada na ferramenta corporativa (*Issue*, *Wiki* ou *Commit* do GitLab), ela não existe oficialmente.

Case Real: A Tragédia Espacial de 1999

Sonda Mars Climate Orbiter (NASA)

A NASA perdeu uma missão interplanetária inteira. O defeito não foi erro de propulsão ou mecânica; o defeito foi uma falha brutal de comunicação entre times de engenharia de software isolados.

- A fabricante do sistema de navegação (Lockheed) programou a força dos propulsores usando a unidade imperial americana (libras-força).
- A equipe de controle (NASA) operou o *software* assumindo que os dados estavam na unidade métrica internacional (Newtons).
- Ninguém exigiu a documentação da API. Ninguém cruzou a informação oficial. O satélite esmagou-se e desintegrou-se na atmosfera marciana. Prejuízo: $125 Milhões de Dólares.

Onde há Prazo, há Atrito

Conflitos em engenharia ágil não são necessariamente ataques ou brigas pessoais. São embates técnicos intensos (Ex: "O código dele é sujo", "Essa API é lenta", "Devemos lançar com bug ou atrasar a sprint?").

O PMBOK mapeia **5 Abordagens Clássicas de Resolução de Conflitos**:

1. **Evitar (Perde-Perde):** O gerente adia o problema porque "não tem tempo de lidar com isso agora". A sujeira é empurrada para debaixo do tapete até que o projeto exploda perto da entrega final.
2. **Apaziguar (Perde-Ganha):** O gerente recua nas suas convicções técnicas arquiteturais apenas para "manter a paz" com o colega. Traz uma ilusão de harmonia no curto prazo, mas gera software frágil e débito técnico no longo prazo.

A Força vs A Concessão

3. **Forçar / Impor (Ganha-Perde):**
   - O Líder usa a autoridade hierárquica ("Faremos do meu jeito porque eu sou o Arquiteto Sênior/Gerente").
   - *Veredito:* Excelente e necessário para situações de altíssima crise (Servidor caiu, precisamos agir em 5 minutos). Péssimo para o dia a dia porque corrói e destrói o moral do time.

4. **Conceder / Reconciliar (Ganha-Ganha parcial):**
   - Cada lado cede um pedaço da sua ideia para fechar um acordo de meio-termo rápido.
   - *Veredito:* Gera soluções sub-ótimas (o projeto vira o "Pato": não corre bem, não nada bem e não voa bem).

5. **Colaborar / Resolver (O Verdadeiro Ganha-Ganha):**
   - **O Padrão Ouro da Gestão Ágil.** A equipe abandona as defesas de ego pessoais.
   - Ambos os lados focam o ataque na **causa raiz** do problema de software (ex: a limitação do banco de dados).
   - A discussão colaborativa é longa e árdua, mas o resultado final é uma **terceira solução técnica** muito superior e mais criativa do que a ideia original de ambos os lados isolados.
   - Exige **altíssima maturidade** dos desenvolvedores para não levarem críticas ao código para o coração.

O Efeito Caixa Preta

Quando construímos software envolvendo Modelos de Linguagem e IA (como na nossa Oficina), lidamos com o perigoso Efeito Caixa Preta na comunicação com o Cliente.

A Ilusão da Magia Ilimitada

O cliente (ou o professor que atua como Sponsor) visualiza ferramentas incríveis fazendo poemas e vídeos online. Como ele não entende a limitação matemática de tokens ou as alucinações estatísticas, ele assume erroneamente que **"com IA, qualquer coisa que eu pedir é rápida, barata e magicamente infalível"**.

Ancoragem de Expectativas Éticas

O cliente subitamente adiciona no *backlog*: *"O Tutor de IA deve prever se o maratonista está com depressão apenas analisando como ele indenta o código"*.

- É seu **dever ético** como Engenheiro de Software ancorar o cliente na realidade empírica.
- Trazer a métrica das limitações do LLM não é "ser negativo" ou "ter má vontade", é uma negociação madura de escopo científico.
- **A Tática Matadora:** Para convencer stakeholders de IA, *nunca use planilhas ou slides*. Crie **protótipos funcionais iterativos** que mostrem a IA travando ou alucinando na prática, assim o cliente entende as fronteiras do seu projeto.

A Comunicação Transparente no GitLab

A melhor estratégia para times remotos ou assíncronos é aplicar a governança de **Transparência Radical** dentro da sua infraestrutura:

- **Status Puxado, e não Empurrado:** O gerente / professor não deve jamais mandar DM privada perguntando "como está?". Ele tem a obrigação de olhar o *Kanban Board* (*Issues Boards*) no GitLab e visualizar as colunas de *Doing* e *Done*.
- **Dúvidas Técnicas Contextualizadas:** Sempre que o código estiver com erro, use a marcação `@usuario` da equipe no campo de comentários dentro da própria *Issue*, anexando os *logs* de erro. Isso cria a "Wiki Orgânica". O WhatsApp jamais faria isso.

Check 02: O Fechamento do Preditivo

Na próxima semana, encerraremos 100% de todo o esforço de Gestão Tradicional (Cascata/PMBOK) da disciplina com o fechamento do Check 02.

A Checklist Final para o Professor (O Sponsor):

1. **Escopo (EAP) e Tempo (Cronograma):** A WBS está documentada na Wiki? As tarefas estão atreladas a *Issues* no Milestone correto e com tempos estimados (`/estimate`) preenchidos?
2. **Qualidade e RH:** Matriz RACI perfeitamente executada (O *Assignee* e os *Reviewers* oficiais do *Merge Request* não se sobrepõem)? O DoD está implementado (Pipeline verde rodando)?
3. **Riscos e Docs:** Pelo menos 5 ameaças ativas em forma de *Issues* com *Labels* de Risco? *ADRs* e plano de arquitetura vivos no README.md?

Reflexões Finais de Arquitetura Social

Antes de fechar os portões da gestão em cascata, responda:

1. Usando a lei matemática da complexidade, se o seu grupo de 4 engenheiros colocar o professor (Sponsor) e um beta-tester no grupo do Telegram (total de 6 pessoas), exatamente quantos novos *Canais de Comunicação* cruzados se abriram simultaneamente nas suas costas?
2. Como o uso agressivo da Matriz de Poder vs. Interesse protege a sua equipe contra a terrível *Microgestão* feita por diretores que deveriam, tecnicamente, apenas figurar no quadrante de "Manter Satisfeitos"?
3. Por que adotar a estratégia de "Apaziguar" para resolver um conflito sobre "qual arquitetura de banco de dados usar" manterá a equipe sorridente no curto prazo, mas destruirá o software de forma invisível nos meses seguintes?

O Repositório (Check 02) está blindado?

Email: alessio@cefetmg.br
Horário de aula: Segunda, 7h às 8:40h
Horário de atendimento: Quarta 16h e whatsapp

Não esqueça de responder o questionário final de módulo!

Próxima fase: Fim definitivo do PMBOK. Entraremos de cabeça no Universo do Desenvolvimento Iterativo, **Metodologias Ágeis e Scrum**.