1 – Uso do Redmine como ferramenta de controle:

Integração com o Git: Utiliza o Git para controle de versões e de releases dos sistemas desenvolvidos, sendo fundamental a integração com os repositórios de forma a ser possível acessar o código fonte gerado no projeto;

Rastreabilidade bidirecional de requisitos: Identificar cada item desenvolvido presente no Documento de Visão do projeto (mais alto nível de abstração), as histórias de usuários do Product Backlog associadas ao item de visão, as tarefas associadas às histórias de usuários e os commits no repositório e vice-versa, possibilitando navegar entre os diversos níveis de atividades, fornecendo ampla visibilidade do projeto;

Histórico de atualização de tarefas: manter um histórico que permitisse visualizar quem alterou o estado da atividade e quando foi realizada a alteração;

Controle da equipe: Configurar a equipe, gerenciar membros de projetos e permissões de acesso, de forma a limitar o acesso a determinadas áreas do sistema para usuários;

Self-hosted: Implementara a ferramenta nos servidores locais, não dependendo de serviços em nuvem para seu funcionamento. Essa não dependência é explícita em alguns contratos com clientes.

2 - O Github como repositório de código-fonte.

A organização das branches estabelece um padrão de nomes e funções para cada tipo de branch, são eles:

master: contém o código de produção, todo o código que está sendo desenvolvido, em algum momento será “juntado” com essa branch.

develop: contém o código do próximo deploy, isso significa que conforme as features vão sendo finalizadas elas vão sendo juntadas nessa *branch* para posteriormente passarem por mais uma etapa antes de ser juntada com a **master.**

feature/*: são *branches* para o desenvolvimento de uma funcionalidade específica, por convenção elas tem o nome iniciado por **feature/**, por exemplo: feature/cadastro-usuarios. Importante ressaltar que essas *branches* são criadas **sempre** à partir da **branch develop**

hotfix/*: são *branches* responsáveis pela realização de alguma correção crítica encontrada em produção e por isso são criadas à partir da **master**. Importante ressaltar que essa *branch* deve ser juntada tanto com a **master** quanto com a **develop.**

release/*: tem uma confiança maior que a branch **develop** e que se encontra em nível de preparação para ser juntada com a **master** e com a **develop** (caso alguma coisa tenha sido modificada na branch em questão).

3. O Google Docs como repositório dos artefatos de especificação.

Criar e aditar documentos em conjunto:

Google Docs é o foco que ele possui na colaboração. Duas ou mais pessoas podem abrir um mesmo arquivo em conjunto e visualizar suas edições em tempo real. Dessa forma facilitando toda elaboração dos artefatos de especificação.

Controle de edições de arquivos

O Google Docs oferece ao usuário pleno controle sobre tudo o que é feito em um documento. Qualquer alteração efetuada possui um registro e é possível restaurar tudo a um ponto antes dela