Para construir o produto correto precisamos:

- Garantir que o Product Owner (PO) e desenvolvedores (Devs) tenham a mesma ideia do que precisa ser produzido;
- Produzir especificações precisas para evitar produzir a coisa errada (desperdiçar esforço);
    - Ou seja, precisamos ter uma linguagem em comum para definir o produto, quando geralmente visionários e técnicos usam linguagens diferentes.
- Formas objetivas de delimitar o cumprimento em detalhes de uma funcionalidade, i.e. CRITÉRIOS DE APROVAÇÃO para funcionalidades;
- Documentação que facilite a compreensão do todo e a realização de mudanças;
- As conversas que definem produtos ocorrem em *workshops de especificação*:
    - Os participantes destas conversas são os POs e os Devs, os visionários e os técnicos;
    - O objetivo desta conversa é definir exemplos que capturem o comportamento do sistema;
    - A ferramenta para esta captura é o Gherkin, uma linguagem-ponte entre a linguagem visionária e a técnica.

Um bom cenário Gherkin é livre de Enchimento Técnico desnecessário, bem escrito, preciso, define o vocabulário comum do domínio de negócio que permite que visionários e técnicos tenham o mesmo entendimento do produto, e é composto pelos exemplos que definem o sistema. O cenário é um exemplo concreto que encapsula uma regra de negócio, e.g.:

Conversa:

> “Vamos imaginar que você vai à cidade numa viagem à negócios usando um aplicativo que fornece rotas de transporte público. Quando que você chega lá? Se você é um deixa-pra-lá e chega lá só uma hora antes da sua reunião, você pode não ter tempo o suficiente para checar se está no endereço certo.”
>
> “É, seria bom a gente colocar uma função de geolocalização para que os usuários consigam saber onde estão.”

Aqui está a mesma conversa em Gherkin:

```
DADO    um passageiro que habilitou geolocalização
QUANDO  o passageiro deseja planejar uma viagem
ENTÃO   o ponto de início deve ser definido como sua localização atual
```

Esta sequência é chamada de *Padrão Dado-Quando-Então*. São algumas das palavras que compõem o pequeno vocabulário do Gherkin.

Vocabulário:

Palavras-chave	Definição
Funcionalidade, Característica	É *o quê* queremos que o sistema faça, o que o cenário deseja explicar
Contexto	O “plano de fundo” do cenaŕio, algo que sempre acontece antes do cenário, as pré-condições para ele
Regra	Uma condição específica que precisa ser seguida, uma restrição
Exemplo ou Cenário	Uma situação concreta que o sistema precisa ser capaz de executar, podendo envolver várias funcionalidades e regras
Esquema do Cenário	Um modelo de situação mais genérico que podemos repetir com detalhes diferentes,
Dado/Dada/Dados/ Dadas	São as variáveis que entram num esquema do cenário, a lista de opções para testar o modelo, e.g.: Tipo = (expresso, cappucino, latte)
Quando	São informações adicionais colocadas encima do contexto
Então	É o que acontece, o resultado de tudo que foi definido anteriormente no exemplo
E	É usando quando acontece mais de uma coisa
MAS	É como o E, mas usado para definir uma exceção

```
TIPOS           expresso, capuccino, latte
INSUMOS         água, pó de café, creme

FUNCIONALIDADE          passar um café
CONTEXTO                a cafeteira está ligada e com insumos suficientes
REGRA                   o processo de passar café não pode ser iniciado sem ág
ESQUEMA DE CENÁRIO      passar um café do <tipo>
CENÁRIO                 passar um expresso
QUANDO                  o usuário inicia o processo de passar um expresso
E                       há água suficiente
E                       há pó de café suficiente
ENTÃO                   a cafeteira passa um expresso


FUNCIONALIDADE          informar o usuário de insumos faltando
CONTEXTO                a cafeteira está ligada e com insumos insuficientes
REGRA                   o processo de passar café não pode ser iniciado sem ág
ESQUEMA DE CENÁRIO      passar um café do <tipo>
CENÁRIO                 passar um latte
QUANDO                  o usuário inicia o processo de passar um latte
E                       há água suficiente
E                       há pó de café suficiente
MAS                     não há creme suficiente
ENTÃO                   a cafeteira não passa um latte
E                       a cafeteira exibe a mensagem "RESERVA INSUFICIENTE DE
```

Gherkin é ótima para definir *requisições comportamentais* do sistema (a forma como o sistema deve se comportar em reação a ações de usuários) graças ao seu foco em ações de usuários. Veja como todos os exemplos tratam das intenções do usuário, sem dizer nada sobre o que há debaixo dos panos. Se um cenário Gherkin começa a falar de coisas fora do domínio do visionário (que é o produto e seus usuários) como conexões a banco de dados (que são do domínio do técnico Dev) ou elementos de interface de usuário como botões (que são do domínio do especialista UX/UI), Gherkin está sendo usada errado.

Essa estrutura torna um cenário Gherkin uma especificação executável. É uma linguagem que os clientes podem usar para definir os critérios para que o sistema seja aceitável, e são executáveis porque dada a funcionalidade “Passar um café”, uma base de testes de sistema com o mesmo nome, “Passar um café”, pode ser invocada e todos os testes que se encaixam em passar um café, como testes sobre os tipos diferentes de café, podem ser executados, e o cliente pode ver um sinal de visto bem bonito aparecendo do lado da especificação em Gherkin e da funcionalidade. É uma ponte entre o *domínio de negócio* e o *domínio técnico*, uma corrente que liga discussões, a texto legível por um leigo, aos testes

automatizados; cria uma progressão natural da *ordem certa* das coisas, definir o comportamento do sistema, criar testes que só passam quando esse comportamento é atendido, e aí escrever código até que todos os testes passem, e de quebra cria uma documentação viva útil para fazer o on-boarding de qualquer novo integrante ao projeto.