Prototipação

A prototipagem de software é a atividade de criar protótipos de aplicativos de software, ou seja, versões incompletas do programa de software que está sendo desenvolvido. É uma atividade que pode ocorrer no desenvolvimento de software e é comparável à prototipagem conhecida de outras áreas, como engenharia mecânica ou manufatura.

Um protótipo normalmente simula apenas alguns aspectos e pode ser completamente diferente do produto final.

A prototipagem tem vários benefícios: o designer e o implementador de software podem obter feedback valioso dos usuários no início do projeto. O cliente e o contratante podem comparar se o software feito corresponde à especificação do software, de acordo com a qual o programa de software é construído.

Ele também permite que o engenheiro de software tenha algumas informações sobre a precisão das estimativas iniciais do projeto e se os prazos e marcos propostos podem ser cumpridos com sucesso. O grau de completude e as técnicas utilizadas na prototipagem estão em desenvolvimento e debate desde sua proposta no início da década de 1970.

O propósito de um protótipo é permitir que os usuários do software avaliem as propostas dos desenvolvedores para o projeto do produto final, experimentando-as de fato, em vez de ter que interpretar e avaliar o projeto com base em descrições. A prototipagem de software fornece uma compreensão das funções do software e possíveis ameaças ou problemas.

A prototipagem também pode ser utilizada por usuários finais para descrever e comprovar requisitos que não foram considerados, e que podem ser um fator chave na relação comercial entre desenvolvedores e seus clientes. O design de interação, em particular, faz uso intenso de prototipagem com esse objetivo.

Esse processo contrasta com o ciclo de desenvolvimento monolítico dos anos 1960 e 1970 de construir o programa inteiro primeiro e depois resolver quaisquer inconsistências entre o projeto e a implementação, o que levou a custos de software mais altos e estimativas ruins de tempo e custo.

A abordagem monolítica foi apelidada de técnica "Slaying the (software) Dragon", uma vez que assume que o designer e desenvolvedor de software é um único herói que tem que matar o dragão inteiro sozinho. A prototipagem também pode evitar a grande despesa e a dificuldade de ter que alterar um produto de software acabado.

A prática da prototipagem é um dos pontos que Frederick P. Brooks faz em seu livro de 1975 The Mythical Man-Month e seu artigo de aniversário de 10 anos " No Silver Bullet ".

Um dos primeiros exemplos de prototipagem de software em larga escala foi a implementação do tradutor Ada/ED da NYU para a linguagem de programação Ada. Foi implementado em SETL com a intenção de produzir um modelo semântico executável para a linguagem Ada, enfatizando a clareza do design e da interface do usuário sobre a velocidade e eficiência. O sistema NYU Ada/ED foi a primeira implementação Ada validada, certificada em 11 de abril de 1983.

Esboço

O processo de prototipagem envolve as seguintes etapas:

Identifique os requisitos básicos

Determine os requisitos básicos, incluindo as informações de entrada e saída desejadas. Detalhes, como segurança, geralmente podem ser ignorados.

Desenvolver protótipo inicial

O protótipo inicial é desenvolvido que inclui apenas interfaces de usuário. (Veja Protótipo Horizontal , abaixo)

Análise

Os clientes, incluindo os usuários finais, examinam o protótipo e fornecem feedback sobre possíveis adições ou alterações.

Revisar E Aprimorar O Protótipo

Usando o feedback, tanto as especificações quanto o protótipo podem ser melhorados. A negociação sobre o que está dentro do escopo do contrato/produto pode ser necessária. Se forem introduzidas alterações, pode ser necessário repetir os passos #3 e #4.

Protótipo Horizontal

Um termo comum para um protótipo de interface do usuário é o protótipo horizontal. Ele fornece uma visão ampla de um sistema ou subsistema inteiro, concentrando-se na interação do usuário mais do que na funcionalidade do sistema de baixo nível, como acesso ao banco de dados. Os protótipos horizontais são úteis para:

Confirmação dos requisitos da interface do usuário e escopo do sistema,

Versão de demonstração do sistema para obter buy-in do negócio,

Desenvolva estimativas preliminares de tempo, custo e esforço de desenvolvimento.

Protótipo vertical

Um protótipo vertical é uma elaboração completa aprimorada de um único subsistema ou função. É útil para obter requisitos detalhados para uma determinada função, com os seguintes benefícios:

Design de banco de dados de refinamento ,

Obtenha informações sobre volumes de dados e necessidades de interface do sistema, para dimensionamento de rede e engenharia de desempenho,

Esclareça requisitos complexos analisando a funcionalidade real do sistema.

Tipos

A prototipagem de software tem muitas variantes. No entanto, todos os métodos são de alguma forma baseados em duas formas principais de prototipagem: prototipagem descartável e prototipagem evolutiva.

Prototipagem Descartável

Também chamado de prototipagem fechada. A prototipagem descartável ou rápida refere-se à criação de um modelo que eventualmente será descartado em vez de se tornar parte do software final entregue. Depois que a coleta preliminar de requisitos é realizada, um modelo de trabalho simples do sistema é construído para mostrar visualmente aos usuários como seus requisitos podem parecer quando implementados em um sistema finalizado. É também uma forma de prototipagem rápida.

A prototipagem rápida envolve a criação de um modelo de trabalho de várias partes do sistema em um estágio muito inicial, após uma investigação relativamente curta.

O método utilizado na sua construção costuma ser bastante informal, sendo o fator mais importante a rapidez com que o modelo é fornecido. O modelo torna-se então o ponto de partida a partir do qual os usuários podem reexaminar suas expectativas e esclarecer seus requisitos. Quando esse objetivo é alcançado, o modelo do protótipo é 'jogado fora' e o sistema é formalmente desenvolvido com base nos requisitos identificados.

A razão mais óbvia para usar prototipagem descartável é que ela pode ser feita rapidamente. Se os usuários puderem obter feedback rápido sobre seus requisitos, eles poderão refiná-los no início do desenvolvimento do software. Fazer alterações no início do ciclo de vida do desenvolvimento é extremamente econômico, pois não há nada a ser refeito nesse ponto.

Se um projeto for alterado após uma quantidade considerável de trabalho ter sido feita, pequenas alterações podem exigir grandes esforços para serem implementadas, pois os sistemas de software têm muitas dependências. A velocidade é crucial na implementação de um protótipo descartável, pois com um orçamento limitado de tempo e dinheiro, pouco pode ser gasto em um protótipo que será descartado.

Outro ponto forte da prototipagem descartável é sua capacidade de construir interfaces que os usuários podem testar. A interface do usuário é o que o usuário vê como o sistema e, ao vê-la na frente dele, é muito mais fácil entender como o sistema funcionará.

Afirma-se que a prototipagem rápida revolucionária é uma maneira mais eficaz de lidar com problemas relacionados aos requisitos do usuário e, portanto, um aprimoramento maior da produtividade geral do software.

Os requisitos podem ser identificados, simulados e testados de forma muito mais rápida e barata quando questões de capacidade de evolução, manutenibilidade e estrutura de software são ignoradas. Isso, por sua vez, leva à especificação precisa dos requisitos e à subsequente construção de um sistema válido e utilizável do ponto de vista do usuário, por meio de modelos convencionais de desenvolvimento de software.

Os protótipos podem ser classificados de acordo com a fidelidade com que se assemelham ao produto real em termos de aparência, interação e tempo. Um método de criar um protótipo descartável de baixa fidelidade é a prototipagem em papel . O protótipo é implementado usando papel e lápis e, portanto, imita a função do produto real, mas não se parece com isso.

Outro método para construir facilmente protótipos descartáveis de alta fidelidade é usar um GUI Builder e criar um click dummy , um protótipo que se parece com o sistema de metas, mas não fornece nenhuma funcionalidade.

O uso de storyboards , animatics ou desenhos não é exatamente o mesmo que prototipagem descartável, mas certamente se enquadra na mesma família. Estas são implementações não funcionais, mas mostram como o sistema ficará.

Resumo: Nesta abordagem o protótipo é construído com a ideia de que será descartado e o sistema final será construído do zero. Os passos desta abordagem são:

Escreva os requisitos preliminares

Desenhe o protótipo

O usuário experimenta/usa o protótipo, especifica novos requisitos

Repita se necessário

Escreva os requisitos finais

Prototipagem Evolutiva

A prototipagem evolutiva (também conhecida como prototipagem de prototipagem) é bem diferente da prototipagem descartável . O principal objetivo ao usar a prototipagem evolutiva é construir um protótipo muito robusto de forma estruturada e refiná-lo constantemente. A razão para esta abordagem é que o protótipo evolucionário, quando construído, forma o coração do novo sistema, e as melhorias e outros requisitos serão então construídos.

Ao desenvolver um sistema usando prototipagem evolucionária, o sistema é continuamente refinado e reconstruído.

"... a prototipagem evolutiva reconhece que não entendemos todos os requisitos e constrói apenas aqueles que são bem compreendidos."

Essa técnica permite que a equipe de desenvolvimento adicione recursos ou faça alterações que não puderam ser concebidas durante a fase de requisitos e design.

Para que um sistema seja útil, ele deve evoluir por meio do uso em seu ambiente operacional pretendido. Um produto nunca está "pronto"; está sempre amadurecendo à medida que o ambiente de uso muda... muitas vezes tentamos definir um sistema usando nosso quadro de referência mais familiar - onde estamos agora.

Fazemos suposições sobre a forma como os negócios serão conduzidos e a base tecnológica na qual os negócios serão implementados. Um plano é elaborado para desenvolver a capacidade e, mais cedo ou mais tarde, algo semelhante ao sistema imaginado é entregue.

Os protótipos evolutivos têm uma vantagem sobre os protótipos descartáveis, pois são sistemas funcionais. Embora eles possam não ter todos os recursos planejados pelos usuários, eles podem ser usados temporariamente até que o sistema final seja entregue.

"Não é incomum dentro de um ambiente de prototipagem para o usuário colocar um protótipo inicial para uso prático enquanto espera por uma versão mais desenvolvida... O usuário pode decidir que um sistema 'defeituoso' é melhor do que nenhum sistema."

Na prototipagem evolucionária, os desenvolvedores podem se concentrar em desenvolver partes do sistema que eles entendem, em vez de trabalhar no desenvolvimento de um sistema inteiro.

Para minimizar o risco, o desenvolvedor não implementa recursos mal compreendidos. O sistema parcial é enviado para as instalações do cliente. À medida que os usuários trabalham com o sistema, eles detectam oportunidades para novos recursos e solicitam esses recursos aos desenvolvedores.

Os desenvolvedores então aceitam essas solicitações de aprimoramento junto com as suas próprias e usam práticas sólidas de gerenciamento de configuração para alterar a especificação de requisitos de software, atualizar o design, recodificar e testar novamente.

Prototipagem Incremental

O produto final é construído como protótipos separados. No final, os protótipos separados são mesclados em um design geral. Com a ajuda da prototipagem incremental, o intervalo de tempo entre o usuário e o desenvolvedor de software é reduzido.

Prototipagem Extrema

A prototipagem extrema como processo de desenvolvimento é usada especialmente para o desenvolvimento de aplicativos da web.

Basicamente, ele divide o desenvolvimento web em três fases, cada uma baseada na anterior. A primeira fase é um protótipo estático que consiste principalmente em páginas HTML. Na segunda fase, as telas são programadas e totalmente funcionais usando uma camada de serviços simulada. Na terceira fase, os serviços são implementados.

"O processo é chamado de Extreme Prototyping para chamar a atenção para a segunda fase do processo, em que uma interface de usuário totalmente funcional é desenvolvida com muita pouca consideração pelos serviços além do contrato."

Vantagens

Há muitas vantagens em usar a prototipagem no desenvolvimento de software – algumas tangíveis, outras abstratas.

Tempo e custos reduzidos: A prototipagem pode melhorar a qualidade dos requisitos e especificações fornecidos aos desenvolvedores. Como as mudanças custam exponencialmente mais para serem implementadas à medida que são detectadas posteriormente no desenvolvimento, a determinação antecipada do que o usuário realmente deseja pode resultar em um software mais rápido e mais barato.

Envolvimento do usuário aprimorado e aumentado: A prototipagem requer o envolvimento do usuário e permite que eles vejam e interajam com um protótipo, permitindo que forneçam feedback e especificações melhores e mais completos. A presença do protótipo sendo examinado pelo usuário evita muitos mal-entendidos e falhas de comunicação que ocorrem quando um lado acredita que o outro entendeu o que foi dito. Como os usuários conhecem o domínio do problema melhor do que qualquer pessoa da equipe de desenvolvimento, o aumento da interação pode resultar em um produto final com maior qualidade tangível e intangível. É mais provável que o produto final satisfaça o desejo do usuário por aparência, toque e desempenho.

Desvantagens

Usar, ou talvez usar mal, a prototipagem também pode ter desvantagens.

Análise insuficiente: O foco em um protótipo limitado pode distrair os desenvolvedores de analisar adequadamente o projeto completo. Isso pode levar ao esquecimento de soluções melhores, à preparação de especificações incompletas ou à conversão de protótipos limitados em projetos finais mal projetados e difíceis de manter .

Além disso, como um protótipo é limitado em funcionalidade, ele pode não ser bem dimensionado se o protótipo for usado como base de uma entrega final, o que pode não ser percebido se os desenvolvedores estiverem muito focados em construir um protótipo como modelo.

Confusão do usuário de protótipo e sistema finalizado: Os usuários podem começar a pensar que um protótipo, destinado a ser jogado fora, é na verdade um sistema final que apenas precisa ser finalizado ou polido. (Eles, por exemplo, muitas vezes desconhecem o esforço necessário para adicionar verificação de erros e recursos de segurança que um protótipo pode não ter.)

Isso pode levá-los a esperar que o protótipo modele com precisão o desempenho do sistema final quando isso não for possível. a intenção dos desenvolvedores. Os usuários também podem se apegar a recursos que foram incluídos em um protótipo para consideração e depois removidos da especificação para um sistema final. Se os usuários puderem exigir que todos os recursos propostos sejam incluídos no sistema final, isso pode levar a conflitos.

Incompreensão do desenvolvedor sobre os objetivos do usuário: Os desenvolvedores podem presumir que os usuários compartilham seus objetivos (por exemplo, entregar a funcionalidade principal no prazo e dentro do orçamento), sem entender questões comerciais mais amplas.

Por exemplo, representantes de usuários que participam de eventos de software empresarial (por exemplo , PeopleSoft) podem ter visto demonstrações de "auditoria de transações" (onde as alterações são registradas e exibidas em uma exibição de grade diferente) sem serem informados de que esse recurso exige codificação adicional e geralmente requer mais hardware para lidar com acessos extras ao banco de dados.

Os usuários podem acreditar que podem exigir auditoria em todos os campos, enquanto os desenvolvedores podem pensar que isso é recursos porque eles fizeram suposições sobre a extensão dos requisitos do usuário. Se o desenvolvedor comprometeu a entrega antes que os requisitos do usuário fossem revisados, os desenvolvedores estão entre uma rocha e um lugar difícil, principalmente se o gerenciamento de usuários derivar alguma vantagem de sua falha na implementação dos requisitos.

Apego do desenvolvedor ao protótipo: Os desenvolvedores também podem se apegar a protótipos que gastaram muito esforço para produzir; isso pode levar a problemas, como tentar converter um protótipo limitado em um sistema final quando ele não possui uma arquitetura subjacente apropriada. (Isso pode sugerir que a prototipagem descartável, em vez da prototipagem evolutiva, deve ser usada.)

Excesso de tempo de desenvolvimento do protótipo: Uma propriedade chave para a prototipagem é o fato de que ela deve ser feita rapidamente. Se os desenvolvedores perderem de vista esse fato, eles podem muito bem tentar desenvolver um protótipo muito complexo.

Quando o protótipo é descartado, os requisitos desenvolvidos com precisão que ele fornece podem não gerar um aumento suficiente na produtividade para compensar o tempo gasto no desenvolvimento do protótipo. Os usuários podem ficar presos em debates sobre detalhes do protótipo, atrapalhando a equipe de desenvolvimento e atrasando o produto final.

Despesa de implementação de prototipagem: os custos iniciais para construir uma equipe de desenvolvimento focada em prototipagem podem ser altos. Muitas empresas têm metodologias de desenvolvimento, e alterá-las pode significar retreinamento, reequipamento ou ambos. Muitas empresas tendem a apenas começar a prototipagem sem se preocupar em retreinar seus funcionários tanto quanto deveriam.

Um problema comum com a adoção da tecnologia de prototipagem são as altas expectativas de produtividade com esforço insuficiente por trás da curva de aprendizado. Além do treinamento para o uso de uma técnica de prototipagem, muitas vezes há uma necessidade negligenciada de desenvolver uma estrutura subjacente específica corporativa e de projeto para suportar a tecnologia. Quando essa estrutura subjacente é omitida, muitas vezes pode resultar em menor produtividade.