Prototipação de Software

- Desenvolvimento rápido de software para validar requisitos

Objetivos

- Descrever o uso de protótipos em diferentes tipos de projetos de desenvolvimento
- Discutir a prototipação evolucionária e descartável
- Apresentar três técnicas de prototipação rápida – desenvolvimento com linguagem de alto nível, programação de banco de dados e reuso de componentes
- Explicar a necessidade de prototipação de interface com o usuário

Tópicos abordados

- Prototipação no processo de software
- Técnicas de prototipação
- Prototipação de interface com o usuário

Prototipação de sistemas

- Prototipação é o desenvolvimento rápido de um sistema
- No passado, o sistema desenvolvido era normalmente considerado inferior ao sistema exigido. Portanto, era necessário desenvolvimento adicional
- Agora, a fronteira entre prototipação e desenvolvimento normal de sistemas é nebulosa e muitos sistemas são desenvolvidos usando uma abordagem evolucionária

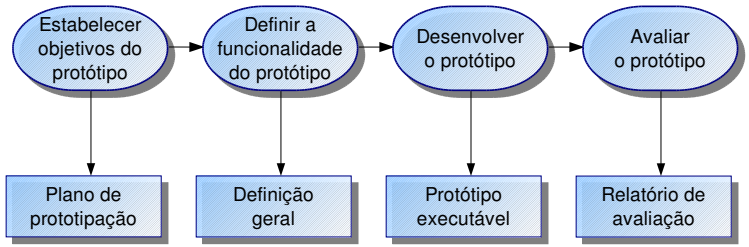
Usos de protótipos de sistemas

- O uso principal é ajudar clientes e desenvolvedores a compreender os requisitos do sistema
  - Elicitação de requisitos. Os usuários podem experimentar com um protótipo para ver como o sistema apóia o seu trabalho
  - Validação de requisitos. O protótipo pode revelar erros e omissões nos requisitos
- A prototipação pode ser considerada como uma atividade de redução de risco, que reduz os riscos com os requisitos

Benefícios da prototipação

- Mal-entendidos entre usuários e desenvolvedores são expostos
- Serviços omitidos podem ser detectados e serviços confusos podem ser identificados
- Um sistema funcional está disponível cedo no processo
- O protótipo pode servir como uma base para derivar uma especificação do sistema
- O sistema pode apoiar o treinamento dos usuários e os testes do sistema

O processo de prototipação



Benefícios da prototipação

- Usabilidade do sistema melhorada
- Alcance mais próximo das necessidades do sistema
- Qualidade do projeto melhorada
- Manutenibilidade melhorada
- Esforço geral de desenvolvimento reduzido

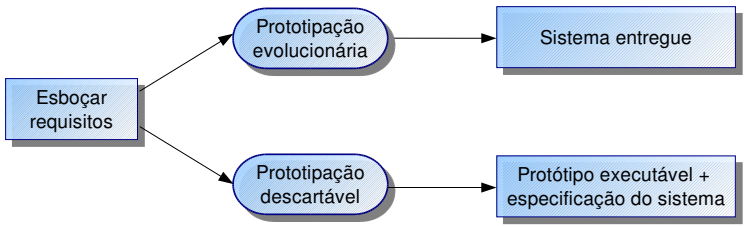
Prototipação no processo de software

- Prototipação evolucionária
  - Uma abordagem de desenvolvimento de sistemas onde um protótipo inicial é produzido e refinado por meio de uma série de estágios até o sistema final
- Prototipação descartável
  - Um protótipo – que é normalmente uma implementação prática do sistema – é produzido para ajudar a descobrir problemas nos requisitos e depois descartado. O sistema é então desenvolvido usando algum outro processo de desenvolvimento

Objetivos da prototipação

- O objetivo da *prototipação evolucionária* é entregar um sistema funcional para os usuários finais. O desenvolvimento inicia com aqueles requisitos que estão melhor compreendidos.
- O objetivo da *prototipação descartável* é validar ou derivar os requisitos do sistema. O processo de prototipação inicia com aqueles requisitos que estão menos compreendidos

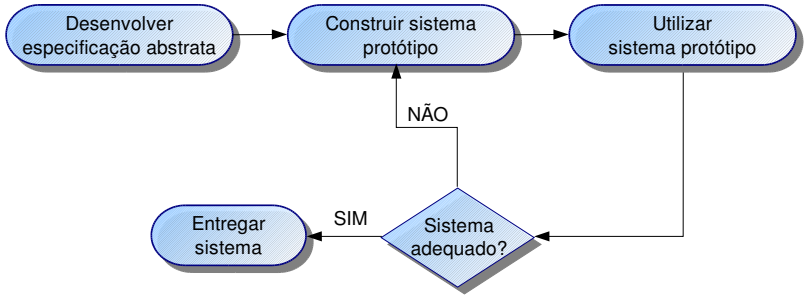
Abordagens da prototipação



Prototipação evolucionária

- Deve ser usado para sistemas nos quais a especificação não pode ser desenvolvida antecipadamente. Por exemplo, sistemas de interface com o usuário de sistemas de IA
- Baseia-se em técnicas que permitem rápidas iterações de sistema
- A verificação é impossível, uma vez que não existe especificação. Validar significa demonstrar a adequação do sistema

Prototipação evolucionária



Vantagens da prototipação evolucionária

- Entrega acelerada do sistema
  - Entrega e desenvolvimento rápidos algumas vezes são mais importantes que a funcionalidade ou manutenibilidade de software no longo prazo
- Compromisso do usuário com o sistema
  - Não apenas é mais provável que o sistema satisfaça os requisitos do usuário, mas também o usuário provavelmente terá um comprometimento maior no uso do sistema

Prototipação evolucionária

- Especificação, projeto e implementação são intercaladas
- O sistema é desenvolvimento como uma série de incrementos que são entregues para o cliente
- São usadas técnicas para desenvolvimento rápido de sistemas, tais como ferramentas CASE e 4GLs
- Interfaces com o usuário são normalmente desenvolvidas usando um toolkit de desenvolvimento de GUI

Problemas da prototipação evolucionária

- Problemas de gerenciamento
  - Processos de gerenciamento existentes assumem o modelo cascata de desenvolvimento
  - São exigidas habilidades de especialistas que podem não estar disponíveis em todas as equipes de desenvolvimento
- Problemas de manutenção
  - Mudança contínua tende a corromper a estrutura do sistema, de tal forma que a manutenção no longo prazo se torna cara
- Problemas contratuais

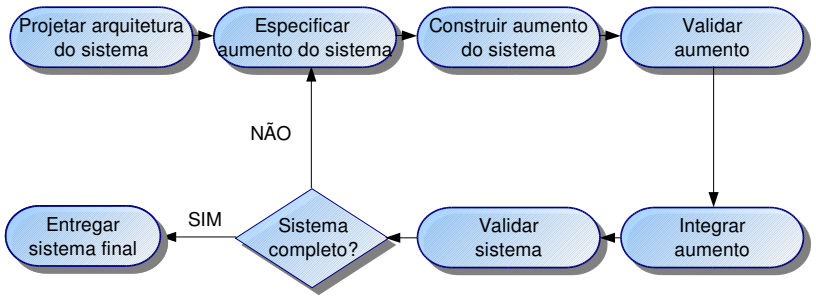
Protótipos como especificações

- Algumas partes dos requisitos (p.ex., funções de segurança crítica) podem ser impossíveis de prototipar e, portanto, não aparecerão na especificação
- Uma implementação não tem o mesmo suporte legal que um contrato
- Os requisitos não funcionais não podem ser adequadamente testados em um protótipo do sistema

Desenvolvimento incremental

- O sistema é desenvolvido em incrementos, após o estabelecimento de uma arquitetura global
- Podem ser desenvolvidos requisitos e especificações para cada incremento
- Os usuários podem experimentar os incrementos entregues enquanto outros estão sendo desenvolvidos. Portanto, isso serve como uma forma de prototipar o sistema
- Pretende combinar algumas vantagens da prototipação com um processo mais gerenciável e melhor estrutura do sistema

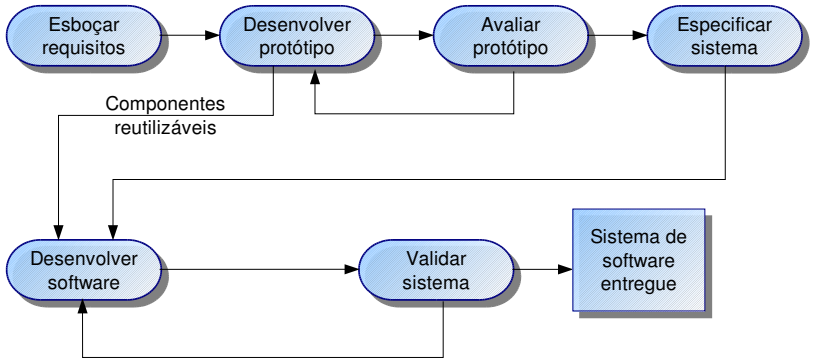
Processo de desenvolvimento incremental



Prototipação descartável

- Usada para reduzir os riscos nos requisitos
- O protótipo é desenvolvido a partir de uma especificação inicial, entregue para teste e então descartada
- O protótipo descartável NÃO deve ser considerado como um sistema final
  - Algumas características do sistema podem ter sido deixadas de fora
  - Não há especificação para manutenção a longo prazo
  - O sistema pode ser fracamente estruturado e difícil de manter

Prototipação descartável



Entrega do protótipo

- Os desenvolvedores podem ser pressionados a entregar um protótipo descartável como um sistema final
- Isso não é recomendado
  - Pode ser impossível ajustar o protótipo para satisfazer os requisitos não funcionais
  - O protótipo inevitavelmente não possui documentação
  - A estrutura do sistema sofrerá degradação com as mudanças feitas durante o desenvolvimento
  - Os padrões de qualidade normais da organização podem não ter sido aplicados

Técnicas de prototipação rápida

- Diversas técnicas podem ser usadas para desenvolvimento rápido
  - Desenvolvimento dinâmico em linguagem de alto nível
  - Programação de banco de dados
  - Montagem de componentes e aplicações
- Essas técnicas não são exclusivas, elas freqüentemente são usadas juntas
- A programação visual é uma parte inerente da maioria dos sistemas de desenvolvimento de protótipos

Linguagens dinâmicas de alto nível

- São linguagens que incluem recursos poderosos de gerenciamento de dados
- Necessitam de um sistema grande de apoio em tempo de execução. Não são normalmente usadas para o desenvolvimento de grandes sistemas
- Algumas linguagens oferecem excelentes recursos de desenvolvimento de IU
- Algumas linguagens possuem um ambiente de apoio integrado cujos recursos podem ser usados no protótipo

Linguagens de prototipação

Language	Type	Application domain
Smalltalk	Object-oriented	Interactive systems
Java	Object-oriented	Interactive systems
Prolog	Logic	Symbolic processing
Lisp	List-based	Symbolic processing

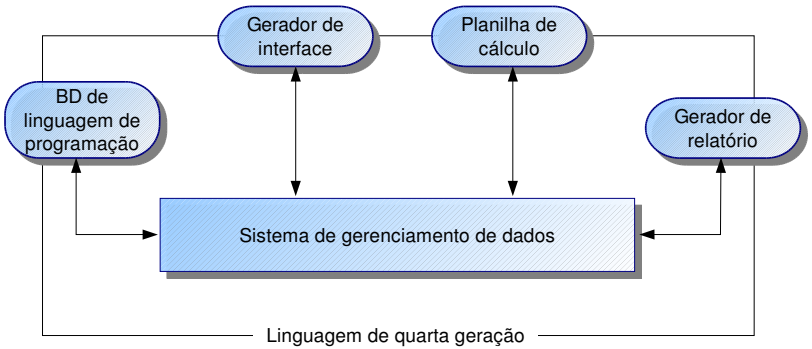
Escolha da linguagem de prototipação

- Qual é o domínio de aplicação do problema?
- Que tipo de interação com o usuário é exigida?
- Que ambiente de suporte acompanha a linguagem?
- Partes diferentes do sistema podem ser programadas em linguagens diferentes. Entretanto, existem problemas com a comunicação entre as linguagens

Linguagens de programação de banco de dados

- Linguagens específicas para sistemas de negócio que se baseiam em um SGBD
- Normalmente, incluem uma linguagem de consulta à base de dados, um gerador de telas, um gerador de relatórios e uma planilha de cálculo
- Podem ser integradas com um conjunto de ferramentas CASE
- A linguagem+ambiente é conhecido às vezes como uma linguagem de quarta geração (4GL)
- Boa relação custo/benefício para sistemas de negócios pequenos a médios

Programação de banco de dados



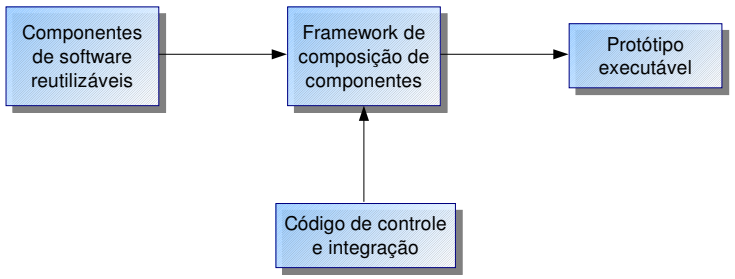
Montagem de componentes e aplicações

- Os protótipos podem ser criados rapidamente a partir de um conjunto de componentes reusáveis, acrescidos de um mecanismo para "colar" esses componentes
- O mecanismo de composição deve incluir recursos de controle e um mecanismo para a comunicação entre os componentes
- A especificação do sistema deve levar em conta a disponibilidade e funcionalidade de componentes existentes

Prototipação com reuso

- Desenvolvimento em nível de aplicação
  - Aplicações inteiras são integradas com o protótipo, de tal maneira que a sua funcionalidade possa ser compartilhada
  - Por exemplo, se é necessário recurso de elaboração de textos, um processador de textos padrão pode ser usado
- Desenvolvimento em nível de componente
  - Componentes individuais são integrados dentro de um framework padrão para implementar o sistema
  - O framework pode ser uma linguagem de script ou um framework de integração, tal como CORBA

Composição de componentes reusáveis



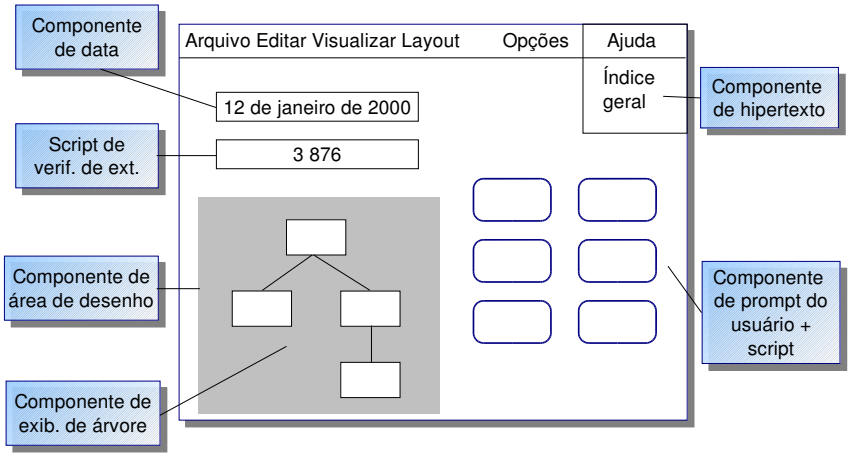
Documentos compostos

- Para algumas aplicações, um protótipo pode ser criado pelo desenvolvimento de um documento composto
- É um documento com elementos ativos (como uma planilha de cálculo, por exemplo) que permite computações do usuário
- Cada elemento ativo possui uma aplicação associada, que é invocada quando aquele elemento é selecionado
- O documento é o próprio integrador de diferentes aplicações

Programação visual

- Linguagens de script, tais como Visual Basic, suportam programação visual, onde o protótipo é desenvolvido pela criação de uma interface com o usuário a partir de itens padrão e associação de componentes com esses itens
- Existe uma grande biblioteca de componentes para apoiar este tipo de desenvolvimento
- Eles podem ser ajustados para suprir os requisitos específicos da aplicação

Programação visual com reuso



Problemas com o desenvolvimento visual

- Difícil de coordenar o desenvolvimento baseado em equipes
- Não há uma arquitetura explícita do sistema
- Dependências complexas entre partes do programa podem gerar problemas de manutenibilidade

Prototipação de interface com o usuário

- É impossível de pré-especificar a apresentação de uma interface com o usuário de uma forma efetiva. Prototipação é essencial
- Desenvolvimento de IU consome uma parte crescente dos custos globais de desenvolvimento de sistemas
- Geradores de IU podem ser usados para “desenhar” a interface e simular a sua funcionalidade com componentes associados a entidades da interface
- Interfaces Web podem ser prototipadas usando um editor de web sites

Pontos principais

- Um protótipo pode ser usado para dar aos usuários finais uma impressão concreta dos recursos do sistema
- A prototipação está se tornando progressivamente mais usada para desenvolvimento de sistemas onde o desenvolvimento rápido é essencial
- Prototipação descartável é usada para compreender os requisitos do sistema
- Na prototipação evolucionária, o sistema é desenvolvido pela evolução de uma versão inicial até a versão final

Pontos principais

- O desenvolvimento rápido dos protótipos é essencial. Isso pode exigir deixar funcionalidades de fora ou relaxar requisitos não funcionais
- Técnicas de prototipação incluem o uso de linguagens de altíssimo nível, programação de banco de dados e construção de protótipos a partir de componentes reusáveis
- A prototipação é essencial para partes do sistema tais como a IU, que não podem ser efetivamente pré-especificadas. Os usuários devem estar envolvidos na avaliação do protótipo