VOLTAR

Padrões de projeto e framework

Apresentar os conceitos de frameworks e padrões de projeto, e diferenciar o uso de cada um deles.

NESTE TÓPICO

Frameworks e padrões de projeto são dois termos muito usados atualmente, mas o caminho que levou a isso é um tanto obscuro para a maioria das pessoas. Neste capítulo, vamos destacar um pouco do histórico que levou à criação e uso dos conceitos de frameworks e padrões de projeto.

Evolução do Desenvolvimento de Software

Quando falamos de desenvolvimento de software, conseguimos retornar ao ano de 1948, na Universidade de Manchester, onde Tom Kilburn, Frederic Williams e Geoff Tootill desenvolveram e executaram um programa de computador para calcular o fator mais alto para o número inteiro 262144 (2 ^8). Esse programa recebeu o nome de “Small Scale Experimental Machine”, que em tradução livre significa Máquina Experimental de Pequena Escala. A ideia desse aplicativo era a de dividir 262144 pelo maior divisor possível, para que o resto da divisão fosse zero, e o resto era subtraído de 262144, e isso foi repetido várias vezes até o resto não pudesse ser mais divisível. O grande diferencial desse programa é que ele é considerado como sendo o primeiro programa de computador a ser armazenado e executado na memória do computador.

Para construir seu programa, os pesquisadores de Manchester se basearam num artigo denominado “A Mathematical Theory of Communication”, que pode ser traduzido por algo como Teoria Matemática da Comunicação, publicado nesse mesmo ano por Claude Shannon, que descrevia um método para escrever uma lógica binária para programar um computador, em

substituição aos métodos eletromecânicos usados até então. O exemplo mais conhecido dessa, que foi usado em alguns lugares até a década de 1980, é o cartão perfurado.

Até a publicação do trabalho de Kilburn, apenas os grandes fabricantes de equipamentos, como a IBM, produziam software, de acordo com seus próprios procedimentos e técnicas. Mas o trabalho de Kilburn, associado a novos tipos de computadores, levou às primeiras empresas independentes de desenvolvimento de software na década de 1950. Essas empresas eram chamadas independentes pois elas não produziam o hardware em que suas aplicações eram executadas, que era a prática comum.

Durante a década de 1950, começou a ocorrer a popularização do uso de computadores, devido ao lançamento dos computadores pessoais, que podiam ser usados nas residências ou em empresas de pequeno e médio porte, ao contrário do modelo vigente até então, com os mainframes, que eram usados apenas dentro do ambiente de trabalho das grandes corporações.

O resultado disso foi a criação de um segmento de mercado, em que houve a massificação de softwares criados com a finalidade de automatizar os processos seguidos pelas empresas, e consequentemente, percebeu-se que muitos processos eram similares, e o resultado, é que os programas de computador acabavam por serem similares. Logo, os desenvolvedores perceberam que poderiam criar seus programas mais rapidamente, se usassem partes de software que já haviam sido desenvolvidos e que já estavam sendo usados.

Isso levou ao conceito de reuso de software, prática que se tornou rapidamente difundida entre os desenvolvedores, mas que também causou alguns problemas, e o modo de trabalho dos desenvolvedores ficou dividido entre aqueles que poderiam se basear em soluções já existentes para resolver novos problemas, e aqueles que preferiam sempre desenvolver seus aplicativos sempre do início, sem reusar nenhum software previamente desenvolvido. [4] define esse segundo modo de trabalho como a “contínua reinvenção da roda” para resolver problemas que já são conhecidos.

A difusão das técnicas de análise e programação orientada a objetos, conforme comentada por [5], explica que o reuso de software, mais do que desejável, é necessário quando se deseja ter uma garantia maior de alcançar um produto final com um nível mínimo de qualidade. Entre as principais ferramentas para o reuso de software estão o uso de frameworks e de padrões de projeto, e conforme pode ser observado no vídeo 1, onde um protótipo de uma aplicação web é criado a partir de um banco de dados já existente, usando frameworks.

Para isso é usada uma IDE que permite configurar um projeto de desenvolvimento baseado em dois frameworks para gerenciar bases de dados e para criação de projetos baseado em arquitetura em 3 camadas. Perceba o quanto isso pode ser vantajoso, se usado corretamente.

Framework

Os frameworks foram desenvolvidos como uma ferramenta que **orienta o desenvolvedor de software a usar certos procedimentos para criar um software**. Ao usar a orientação a objetos, é mais comum criar a estrutura em que o aplicativo será desenvolvido nos conceitos de abstração, polimorfismo e herança.

Assim, os frameworks costumam estruturar o código-fonte usado para criar um aplicativo em classes funcionais, que vão além da representação da base e dados e da criação de telas para interação com os usuários. Essas classes criam certos tipos de objetos e as interações entre os mesmos, para que uma determinada funcionalidade do software funcione.

O código gerado pelo framework pode ser entendido como um template, a ser customizado pelo desenvolvedor de acordo com a aplicação que ele está desenvolvendo. Dependendo da situação, e do framework usado, é possível reusar subsistemas inteiros, aumentando assim o grau de reutilização e contribuindo para uma melhor qualidade do software.

Um dos primeiros frameworks a ser amplamente difundido na orientação a objetos foram o SmallTalk Model-View-Controller (MVC), que serviu de base para o Vídeo 1, e as principais vantagens de usar um framework são:

- Facilidade de reuso de partes prontas, para implementar novos sistemas;
- Segmentar o processamento no máximo de classes possíveis: o framework se torna mais flexível à medida que quanto mais específicas as classes que

eles implementam , assim, para coletar dados e gravar numa base de dados, é melhor ter uma classe para a tela, outra para representar a tabela na base de dados e outra classe para conectar as outras duas classes do que ter apenas uma classe que faça tudo isso;

- Possibilidade de expandir o sistema: ao usar classes mais específicas, é facilitada a adição de novas partes aos sistemas, através do uso de interfaces que permitam conectar o novo software ao que já existe;

Outra característica dos frameworks que costuma ser apresentado é que os frameworks passam a controlar a execução das classes e do software, retirando parte da responsabilidade do desenvolvedor em criar classes de controle. Isso é um ponto controverso, que alguns apontam como vantagem, e outros apontam como desvantagem. Na prática, há sempre os dois lados da moeda, e cabe ao desenvolvedor analisar a melhor maneira de usar um framework.

Padrões de Projeto

[5] apresenta os padrões de projeto, também conhecidos como *design patterns*, como sendo modelos que orientam o desenvolvedor e implementar uma determinada solução para resolver um problema específico. Num primeiro momento, parece ser o mesmo conceito de framework. Entretanto, apesar da semelhança, há uma diferença básica.

Enquanto um framework orienta o desenvolvedor sobre como estruturar um software que será desenvolvido, um **padrão de projeto orienta o desenvolvedor sobre como implementar uma solução para um problema específico desse software** a ser desenvolvido. Vamos usar como exemplo um sistema de vendas on-line para tentar dissecar a diferença entre ambos os conceitos.

Quando Usar Framework e Padrão de Projetos

Ao analisar de modo genérico os vários sistemas de vendas on-line, podemos destacar que um aplicativo desse tipo necessita basicamente das seguintes funcionalidades:

- Cadastro de produtos;
- Cadastro de clientes;
- Pagamento On-Line;
- Carrinho de compras (ou lista de compras, ou pedido);
- Pesquisa de produtos;

Por ser um sistema on-line, esse software necessita ser desenvolvido em interface web, usar uma base de dados, e ter um modo de interagir com bancos e operadoras de cartão de crédito para confirmar os pagamentos.

Ao usar a orientação a objetos para desenvolver esse software, se costuma usar uma estrutura de software que separe as classes em funções específicas para implementar cada um desses módulos. Para isso, usamos o framework.

Mas ao analisar uma funcionalidade específica desse software, como por exemplo a pesquisa de produtos, é possível perceber que se o sistema de vendas on-line precisa lidar com um volume muito grande de registros, processar essas informações e exibi-las para o usuário pode levar um tempo considerável, devido a um processamento complexo de dados, se o desenvolvedor optar por carregar os resultados de cada página diretamente do servidor de dados, conforme o usuário navega pelas telas com o resultado da pesquisa.

Esse é um problema específico do software, e nesse caso, é recomendado usar um padrão de projeto. No caso específico do processamento desse resultado de pesquisa, há alguns padrões de projeto que orientam o desenvolvedor a manter o resultado da pesquisa na memória, já organizado na forma de páginas, e conforme o usuário navega pelas páginas, o aplicativo lê a parte correspondente à nova página da memória, e a apresenta ao usuário. Como resultado da aplicação desses padrões de projeto, tem-se um processamento mais simples, que demora muito menos tempo para ser executado.

Conclusão

O desenvolvimento de software pode ser facilitado, feito com mais qualidade, com mais eficiência e mais rapidamente se forem usados frameworks e padrões de projeto. Na literatura do desenvolvimento de software, é possível encontrar vários frameworks e padrões de projeto de uso comum, e aplicados de modo bem generalista.

Entretanto, softwares usados em determinados tipos de aplicações, como medicina, na indústria eletrônica, militar, comunicações, jogos, etc... possuem seus próprios frameworks e padrões de projetos, que já orientam o desenvolvedor sobre implementações em situações bastante específicas. Assim, dependendo da sua área de atuação, é recomendável que você tome conhecimento do que é usado, e aplicá-lo sempre que possível.

Além disso, você também pode construir seus próprios frameworks e padrões de projeto, e para isso, deve investir bastante tempo para assegurar a qualidade e eficiência do framework ou padrão de projeto que você está desenvolvendo.

Quiz

Exercício Final

Padrões de projeto e framework

INICIAR

Referências

SHANNON, Claude E. A Mathematical Theory of Communication, The Bell Systems Technical Journal, Vol. 27, pp 379-423, 623-656, July, October, 1948

SIMON, Lavington. A History of Manchester Computers (2nd ed), Swindon: The British Computer Society, 1998

GAMMA, E; HELM, R; JOHNSON, R; VLISSIDES, J. Active Guidance of Framework Development Software - Concepts and Tools. Springer-Verlager, 1995, p. 94-103.

FOWLER, Martin. UML Essencial: Um breve guia para a linguagem-padrão de modelagem de objetos 3. ed. Porto Alegre, Bookman, 2005.

Avalie este tópico

ANTERIOR
Tecnologias de BPM

Índice

PRÓXIMO
Sistemas colaborativos

Biblioteca (https://www.uninove.br/conheca-a-uninove/biblioteca/sobre-a-biblioteca/apresentacao/)
Portal Uninove (http://www.uninove.br)
Mapa do Site

® Todos os direitos reservados

Ajuda? (https://ava.un idCurso=)