ENGENHARIA DE *SOFTWARE* PARA INTELIGÊNCIA ARTIFICIAL

UNIDADE I

ENGENHARIA DE *SOFTWARE*

Elaboração

Diogo Santos Silva da Costa

Atualização

Alexander Francisco Vargas Salgado

Produção

Equipe Técnica de Avaliação, Revisão Linguística e Editoração

SUMÁRIO

CAPÍTULO 1

AMBIENTE DE ENGENHARIA DE *SOFTWARE* E MODELAGEM DE *SOFTWARE*

Desenvolver programas viáveis tanto funcional como economicamente são os principais objetivos das metodologias criadas pela disciplina de engenharia de *software*. Para o desenvolvimento destes programas, criar e gerenciar projetos de *software* bem fundamentados e apoiados nessas técnicas é um desafio que demanda muito estudo e acompanhamento. Para isso se fazer valer, o uso de ambientes que integrem diversas ferramentas que irão apoiar de forma eficiente o projeto do novo *software* é fundamental para garantir que todos os objetivos da fase de desenho e requisitos do *software* sejam atendidos o mais plenamente possível, com maior reutilização do código e índice de manutenibilidade satisfatório.

O índice de conclusão com sucesso no desenvolvimento de novos *softwares* deixa muito a desejar, o levantamento de riscos e a análise de requisitos são os problemas mais enfrentados pela equipe de desenvolvimento logo nas fases embrionárias de um projeto, recaindo não só sobre o gerente do projeto, mas também sobre toda a equipe envolvida. A tarefa de ajustar os processos durante as fases seguintes – quando tardiamente serão na prática percebidas e sofridas, o que nem sempre é possível de ser feito de forma satisfatória –, por muitas vezes acaba gerando um legado nada desejado ao *software* desenvolvido ainda antes do seu lançamento, o qual por muitas vezes é adiado para se refazer todos os ajustes, frequentemente ajustes mínimos, mas que criam atrasam e podem criar programas que já nascem fadados ou ao fracasso comercial ou a um custo de manutenção que muitas vezes pode tornar inviável sua continuidade.

Nesse aspecto, é de grande valia se apoiar em ferramentas que irão maximizar os aspectos gerenciais do projeto em desenvolvimento, assim como elucidar os problemas encontrados durante esse processo.

Criar ambientes de gerenciamento de projeto que usem inclusive as técnicas de Inteligência Artificial, como as Redes Neurais, Lógica Fuzzy, entre outras, para maximizar e antever a solução desses problemas, inclusive se baseando no histórico de desenvolvimento de outros projetos similares tanto no quesito de *software* quanto no aspecto dos recursos da equipe envolvida no projeto é trazer a IA para atuar dentro do projeto de desenvolvimento de novos *softwares*, no entanto esse aspecto fica como curiosidade para vocês pesquisarem como anda o desenvolvimento de ferramentas dessa natureza no mercado de *software*.

Não deixe de visitar os links abaixo para complementar seus estudos:

http://files.engenharia-de-*software*7.webnode.com/200000026-b805cb9010/Intelig%C3%AAncia%20Artificial%20Aplicada%20a%20Ambientes%20de%20Engenharia%20de%20*Software*.pdf

Modelagem de *Software*

Criar *softwares* confiáveis e eficientes que executem em diversos ambientes de hardware e diferentes sistemas operacionais são os principais objetivos dessa área do conhecimento. Para isso, o projeto de desenvolvimento deve se basear num tripé de processos bem definidos e maduros: métodos, ferramentas e procedimentos.

Devemos entender e executar o projeto de um novo *software*, de modo que precisamos estimar os diversos aspectos envolvidos e todos os requisitos funcionais e não funcionais; desenhar a arquitetura necessária dentro daquilo que pretendemos e temos disponível; gerar procedimentos e cargas de teste que sirvam para validar o que está sendo proposto; e propor um plano de manutenção que seja prático, economicamente viável e factivel.

O que irá tornar tudo isso real serão as ferramentas usadas, que devem ser escolhidas ou, em casos específicos, até desenvolvidas – mesmo que raramente aconteça, por já se encontrar ferramentas em abundância disp oníveis no mercado com diversos intuitos para dar suporte eficiente ao desenvolvimento de *softwares*.

Com as sequências de métodos definidas e as ferramentas escolhidas, podemos desenhar com eficácia os procedimentos, isto é, as tarefas em si que devem ser executadas para o novo projeto alcançar o sucesso desejado.

Esses procedimentos definirão o conjunto de etapas necessárias e os chamaremos de ciclo de vida; porém, para tanto, precisamos basear tais procedimentos em fases do processo de desenvolvimento. De forma ampla, estas fases podem ser entendidas em: “O que fazer?” “Como fazer?” e “O que mudar?”, as quais tecnicamente podem ser respondidas

com base nas três grandes áreas de desenvolvimento de *software* conhecidas: Definição, Desenvolvimento e Manutenção.

Na fase de definição, deverão ser tratados aspectos como: análise de requisitos, análise do sistema de *software* e planejamento do projeto de *software*.

Na de desenvolvimento, serão tratados os assuntos relacionados com o projeto em si, a codificação e a realização dos testes.

Por último, e não menos importante, temos a fase de manutenção, na qual serão levantadas e executadas as correções e adaptações necessárias, assim como as melhorias funcionais do *software* em questão.

CAPÍTULO 2

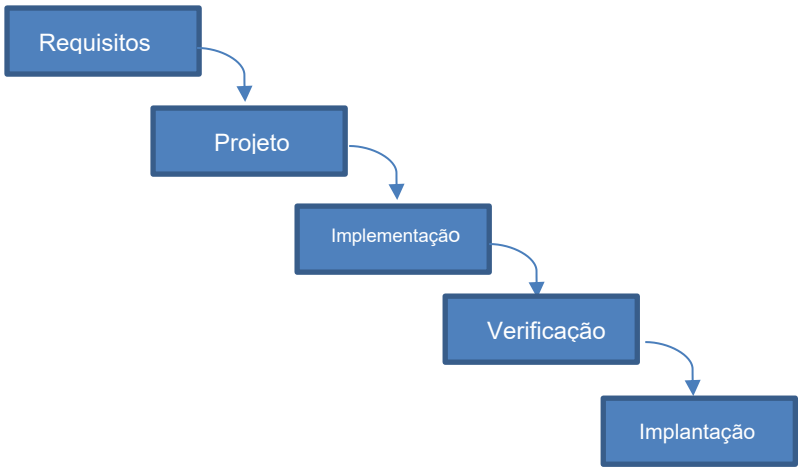
CICLO DE VIDA DE *SOFTWARE*

O tripé que foi apresentado no Capítulo 1 desta unidade, definição, desenvolvimento e matutenção, tem o objetivo de viabilizar as fases do ciclo de vida do *software*: projeto, implementação, avaliação e melhora contínua do *software*. Para isso, uma coleção de técnicas conhecidas e convenções definidas devem ser usadas para apresentar as etapas do ciclo de vida de desenvolvimento de *software*, isto é, um conjunto de práticas estruturadas que tem como objetivo resultar num aumento significativo da qualidade e produtividade do processo de desenvolvimento de *software*.

Os modelos mais difundidos de clico de vida de *software* são cascata, iterativo e incremental e espiral.

O modelo em cascata é definido como uma sequência unidirecional de tarefas que transforma requisitos em funcionalidades de um sistema. É modelado tendo o fim de uma fase o início da próxima, resultando na aparência de uma "cascata" com fases dependentes e interligadas, de modo que são gerados resultados que, ao final do processo, serão juntados para criar o resultado final.

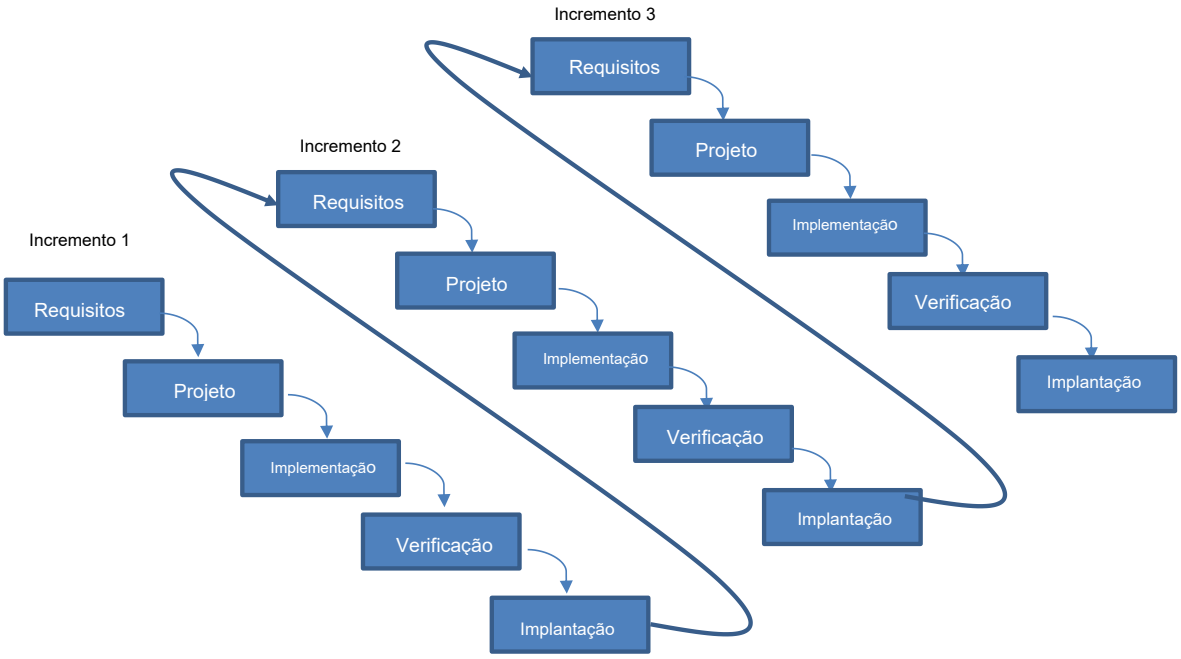
Figura 1. Modelo em Cascata de desenvolvimento de software.



Fonte: Próprio autor, 2020.

Quanto ao modelo iterativo e incremental, tem como resultado uma sequência de versões executáveis do *software*, gerando assim uma integração contínua da arquitetura do sistema para a produção de novas versões.

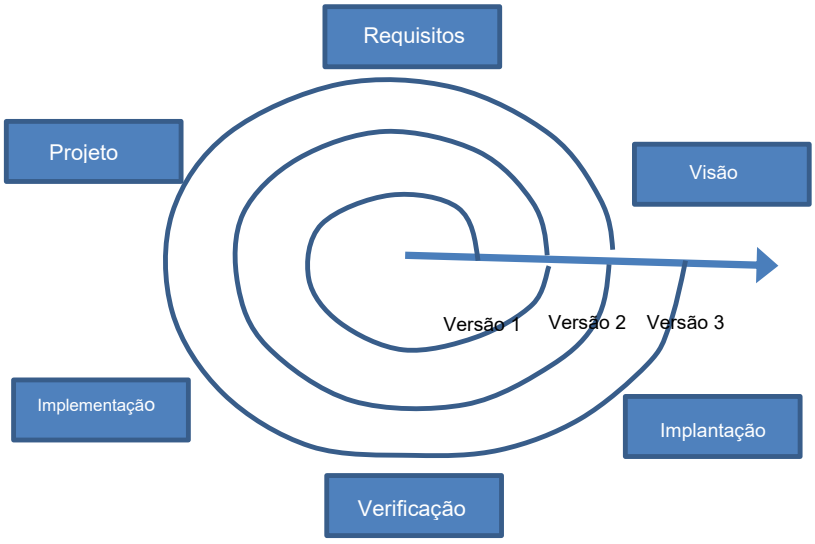
Figura 2. Modelo Iterativo e Incremental.



Fonte: Próprio autor, 2020.

Já o modelo espiral, ou de Boehm, é um modelo cíclico e experimental, que minimiza o grau de risco do projeto e maximiza os de definição e implementação.

Figura 3. Modelo em Espiral, ou de Boehm.



Fonte: Próprio Autor, 2020.

As técnicas mais atuais de engenharia de *software* dão ênfase aos aspectos de reusabilidade e encapsulamento de código testado, diminuindo o tempo de desenvolvimento e aumentando a segurança, o que melhora muito o custo final do projeto.

CAPÍTULO 3

TÉCNICAS DE IA APLICADAS EM ES E EXEMPLOS DE IA APLICADAS EM SOLUÇÕES CONHECIDAS

Agora que sabemos do que a Engenharia de *Software* trata, isto é, os métodos de desenvolvimento de programas de aspecto geral de forma organizada e planejada, voltemos então ao foco principal desta unidade, no qual queremos conhecer: o que há de mais novo em métodos de desenvolvimento de *softwares* para IA; como se deve usar as técnicas conhecidas hoje de engenharia de *software* para apoiar o desenvolvimento de algoritmos e soluções que se baseiam em técnicas de IA.

Hoje já é muito publicado sobre Aprendizado de Máquina (*Machine Learning*), *Deep Learning*, análise preditiva, *Big Data Analytics* e Internet das Coisas (IoT), de tal modo que a inteligência artificial compõe essas tecnologias entre outras mais.

O principal objetivo não é só dar autonomia aos sistemas desenvolvidos com esses algoritmos, mas também produzir conhecimento confiável, de forma extremamente rápida, e em volumes imensos de dados, de modo quase independente a partir de entradas pré-definidas. Muitos profissionais da área trabalham em projetos para fazer com que os computadores sejam capazes também de capturar emoções das pessoas, por meio da análise de imagens, a fim de demonstrar diversas formas de sentimentos antes exclusivos dos humanos.

Esses são recursos já possíveis, porém com claras limitações, mas não deixam de ser polêmicos, como foi o caso da IA da Microsoft, que, após um tempo sendo insultada, aprendeu isso e passou a insultar também os usuários do site.

Vamos então focar em como você pode aproveitar essa tecnologia no desenvolvimento de sistemas. Nesse sentido, podemos partir de dois pontos distintos, podemos iniciar o desenvolvimento de um *software* apoiado nas técnicas de IA, ou podemos incluir essas técnicas como novos recursos em sistemas já existentes. Para isso precisamos, então, conhecer os principais modelos e técnicas usados hoje para treinar máquinas, isto é, os algoritmos de aprendizados em seus diversos aspectos.

O primeiro algoritmo de aprendizado que vamos falar é conhecido como Aprendizado de Máquina, do inglês *Machine Learning*. Esse algoritmo envolve *softwares* usando dados para compreender situações e ambientes com o mínimo de programação possível. Com isso, em vez de programar padrões de funcionamento, o *Machine Learning* consiste em pensar o modo como o sistema irá aprender os padrões necessários de forma autônoma partindo dos dados que recebe e interpreta, atuando sobre os dados das entradas atuais.

Ou seja, o *Machine Learning* é uma tecnologia pertencente ao campo da inteligência artificial dedicada a medir a capacidade que o *software* tem de aprender sozinho, bem como tornar isso possível. Hoje, o aprendizado das máquinas é a principal tecnologia impulsionadora da inteligência artificial.

Você pode ver o *Machine Learning* em ação quando pesquisa um determinado produto na internet e, quando visita uma loja virtual, lá está ele aparecendo em destaque, como recomendável para você. Isso é feito a partir de algoritmos escritos e pensados para adquirir a base de conhecimentos necessários para as decisões, por meio de algoritmos e *big data*, identificando padrões de dados e criando conexões entre eles para aprender a executar uma tarefa sem a ajuda humana e de forma inteligente.

Esses algoritmos usam análises estatísticas para prever respostas mais precisamente e entregam o melhor resultado preditivo com menos chance de erro. Essa tecnologia pode ser separada em duas categorias principais: supervisionada ou não supervisionada.

Os algoritmos supervisionados são aqueles em que o ser humano precisa interagir controlando a saída e entrada de dados e interferir no treinamento da máquina, além de fazer comentários sobre a precisão das previsões. Por fim, a máquina aplica o que foi aprendido no seu algoritmo para a próxima análise.

Já na categoria não supervisionada, os algoritmos utilizam o *deep learning* (aprendizagem profunda) para processar tarefas complexas sem o treinamento humano.

A partir desse conhecimento, ao iniciar o levantamento de requisitos e planejamento de desenvolvimento do novo *software* em questão, que irá adotar as técnicas e modelos de IA em seu escopo, poderemos definir o perfil da equipe que precisaremos montar, profissionais com conhecimento de estatística, banco de dados, além de programação serão de grande valia para maximizar o sucesso da aplicação. Ademais, necessário um projeto de *hardware* bem dimensionado para dar suporte a todo volume de processamento que será necessário para o sucesso da implementação.

Conhecem a polêmica do sistema de IA da Microsoft? Leiam o artigo abaixo!

https://www.tecmundo.com.br/inteligencia-artificial/102835-microsoft-explica-episodio-chatbot-racista-diz-tay-deve-voltar.htm

Conheça mais sobre Inteligência Artificial, *Machine Learning* e *Deep Learning*:

https://rockcontent.com/blog/machine-learning/

Exemplos de IA aplicadas em soluções conhecidas

Google: realiza o preenchimento automático das buscas e prevê com alta precisão o que o usuário deseja pesquisar. O Google também já usa a IA para testar carros 100% autônomos, evitando colisões e engarrafamentos.

Amazon: uma das maiores empresas de *e-commerce* do mundo, faz recomendações personalizadas de produtos e serviços aos usuários usando algoritmos de *Machine Learning*.

Waze: o aplicativo de navegação reúne dados do usuário, na internet, e da região que está inserido, via satélite, para indicar as melhores rotas em apenas um clique.

Facebook: a rede social mais famosa do mundo conta com o reconhecimento facial das imagens postadas para recomendar marcações em fotos.

Siri: o aplicativo de assistente pessoal usa o processamento de voz para se comunicar com o usuário.

REFERÊNCIAS

CORMEN, T. H.; LEISERSON, C. E.; RIVEST, R. L. **Introduction to Algorithms**. 3. ed. Massachusetts, MA: MIT Press, 1.292 p. 2009.

ELKNER, J.; DOWNEY, A.; MEYERS, C. **How to think like a computer scientist**: learning with Python. Wickford, UK: Samurai Media Limited, 306 p. 2016.

GRUS, J. **Data science from scratch**: first principles with python. Sebastopol, CA: O'Reilly, 330 p. 2015.

HERSTEIN, N.; WINTER, D. J. **Matrix Theory and Linear Algebra**. USA: Macmillan Pub. Co., 508 p. 1988.

JOLLY, K. **Hands-on data visualization with Bokeh**: interactive web plotting for Python using Bokeh. Packt Publishing Ltd, 174 p. 2018.

MADHAVAN, S. **Mastering Python for Data Science**. USA: Packt Publishing, 294 p. 2015.

MÜLLER, A. C.; GUIDO, S. **Introduction to machine learning with Python**: a guide for data scientists. 1. ed. O'Reilly Media, 392 p. 2016.

SEDGEWICK, R.; WAYNE, K. **The textbook Algorithms**. 4, USA: Ed. Addison-Wesley Professional, 992 p. 2011.

TATSUOKA, Maurice M.; LOHNES, Paul R. **Multivariate analysis:** Techniques for educational and psychological research. USA: Macmillan Publishing Co. Inc, 479 p. 1988.