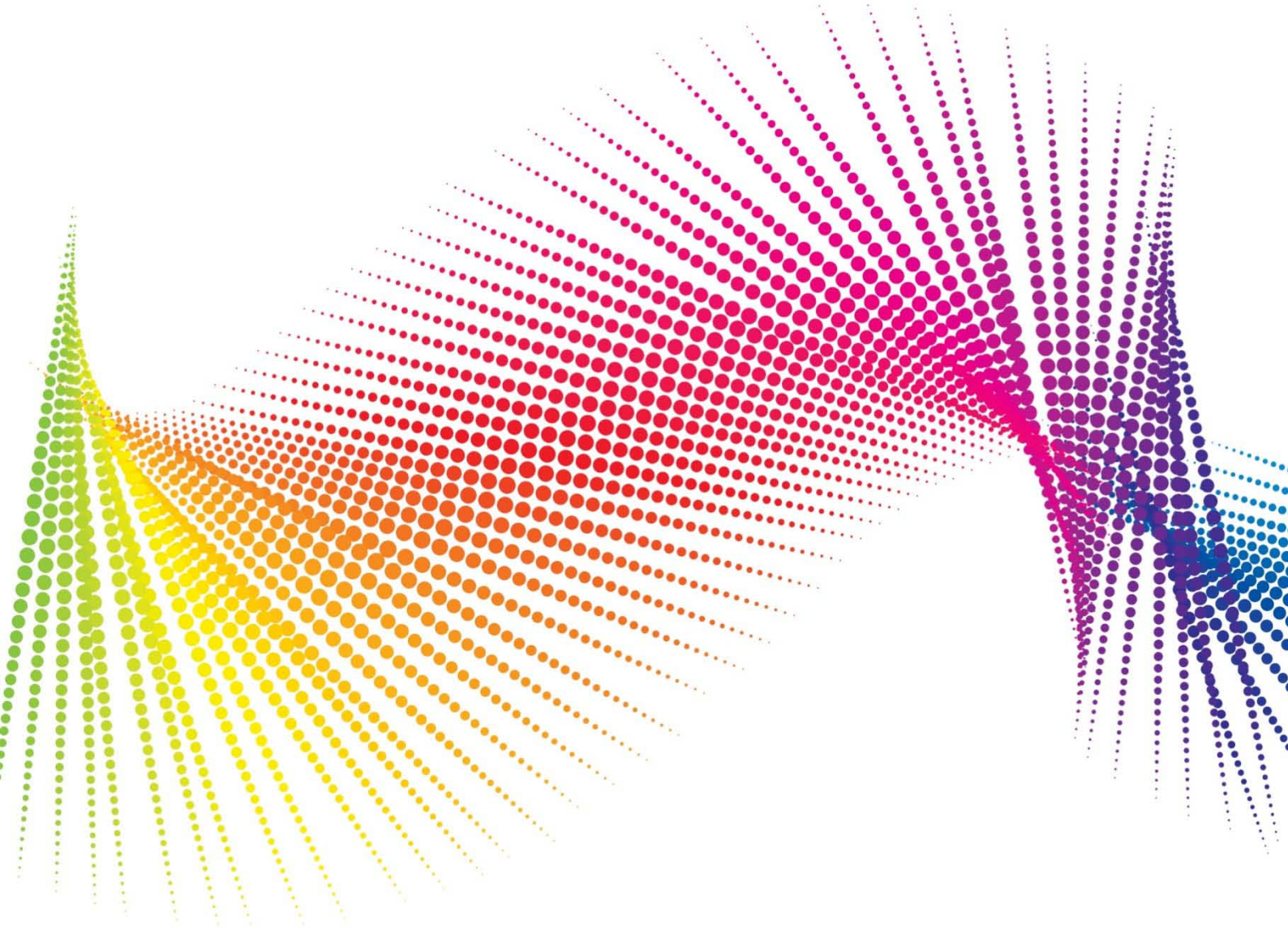
Análise e projeto de sistemas

Aula 03



Este material é parte integrante da disciplina oferecida pela UNINOVE.
O acesso às atividades, conteúdos multimídia e interativo, encontros virtuais, fóruns de discussão e a comunicação com o professor devem ser feitos diretamente no ambiente virtual de aprendizagem UNINOVE.

Uso consciente do papel.
Cause boa impressão, imprima menos.

Aula 03: Ciclos de vida dos sistemas

Objetivo: Apresentar ao aluno os conceitos sobre ciclos de vida dos sistemas de informação, bem como as etapas que os compõem.

Ciclo de vida – definição

Assim como os seres humanos possuem diversas etapas em sua vida (nascimento, crescimento, maturidade e morte), o software também possui etapas semelhantes. Um software, entretanto, pode ser projetado para ser mais eficiente, resistir mais e melhor às mudanças do ambiente que o rodeia, ser mais otimizado e versátil. Para que isso seja possível, é necessário que se adote um método para o seu desenvolvimento, definindo as etapas que deverão ser seguidas, a forma de sua criação, implantação e manutenção.

Ciclo de vida do desenvolvimento de sistemas é o conjunto de etapas ou fases que existem para o desenvolvimento de sistemas aplicativos. Os ciclos de vida foram sendo influenciados pelo surgimento de novas linguagens de programação, por novas técnicas de modelagem de dados, pelas necessidades que dia a dia são modificadas e em função de novas tecnologias. A metodologia aplicada na abordagem de cada etapa do ciclo de vida indica a maleabilidade do processo.

Ciclo de vida em cascata

Também chamado de ciclo de vida clássico, foi a primeira forma de desenvolvimento de sistemas, surgiu no final dos anos 1960 e início dos anos 1970. O principal objetivo da informática naquela época era automatizar os processos existentes. Dessa necessidade, surgiam sistemas isolados que atendiam às necessidades de automatização de áreas específicas.

Nesse ciclo de vida não é criado algum tipo de modelo, não são utilizadas técnicas de estruturação e praticamente não existe oportunidade para o usuário realizar alguma alteração em pontos específicos. Uma vez fechados os requisitos, eles são quase imutáveis. As atividades são realizadas em sequência e não existe

retorno entre as atividades. Toda a documentação, quando feita, é produzida após o término do projeto. O sistema é considerado como uma entidade monolítica, ou seja, não existe o conceito de segmentar o sistema em módulos e, a partir dessa segmentação, tratar cada módulo como um problema menor a ser resolvido.

Fica evidente que os projetos realizados com esse ciclo de vida se caracterizam pela alta incidência de manutenção, pois estão sujeitos a poucas alterações durante o desenvolvimento.

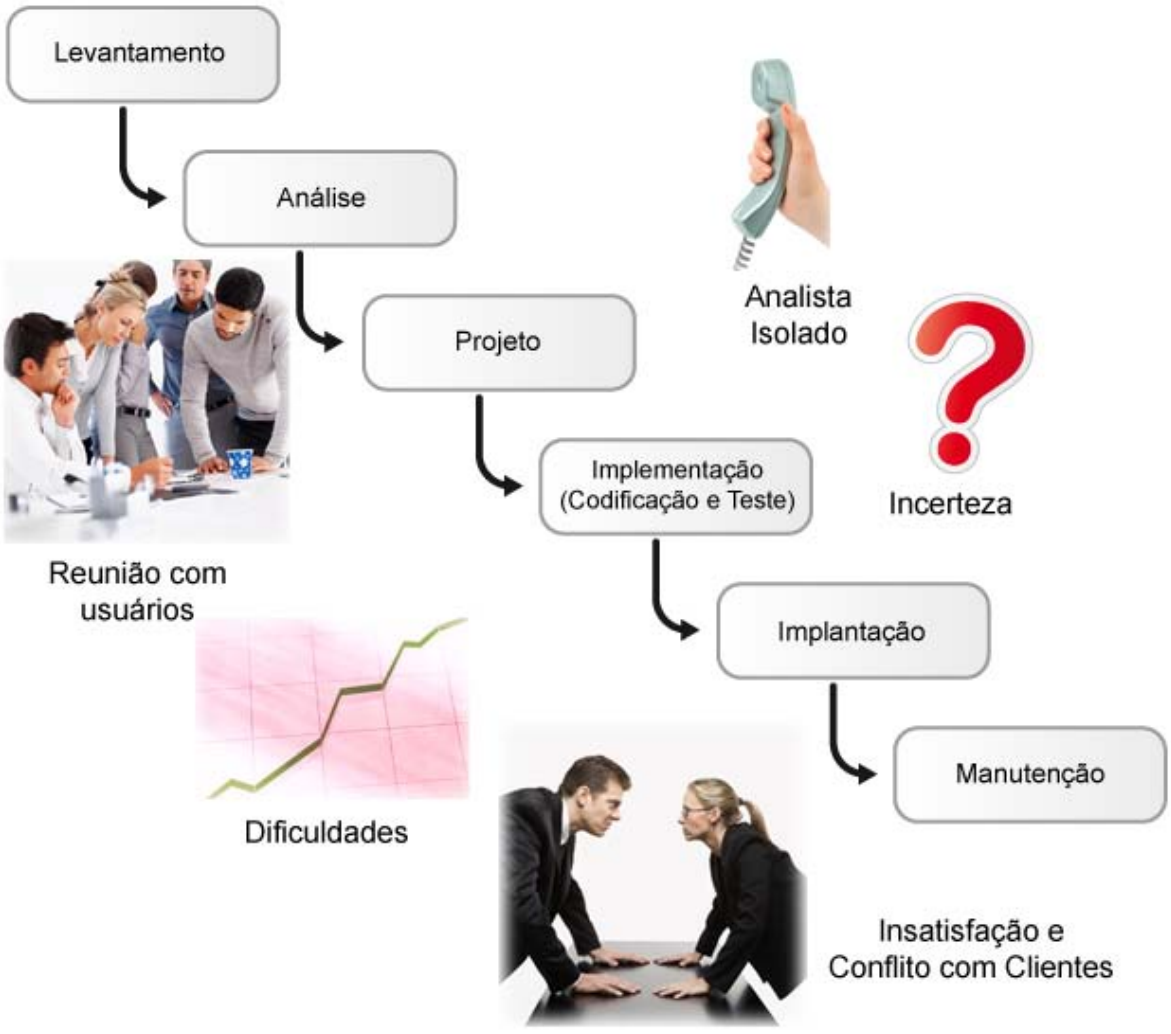


Figura 1 – Ciclo de vida em cascata

Ciclo de vida em espiral

É um tipo de desenvolvimento de sistemas mais maleável e adaptável, se comparado ao ciclo de vida em cascata. Consiste em agregar funcionalidades ao sistema de forma evolutiva, apresentando resultados parciais em relação à meta desejada (final do desenvolvimento do sistema). Os resultados que são apresentados devem ser operacionais, isto é, devem ser implantados para utilização do cliente. Esse tipo de ciclo de vida permite, ainda, que sejam adaptados novos métodos ao longo do desenvolvimento do sistema.

Figura 2 – Ciclo de vida em espiral

Ciclo de vida da prototipação

Muitas vezes, o cliente define um conjunto de objetivos gerais para o software, mas não identifica requisitos de entrada, processamento e saída detalhados. Em outros casos, o desenvolvedor pode não ter certeza da eficiência de

um algoritmo, da adaptabilidade de um sistema operacional ou da forma que a interação homem-máquina deve assumir. Nessas, e em muitas outras ocasiões, uma abordagem de prototipação à engenharia de software pode representar a melhor abordagem.

O modelo pode assumir uma das três formas:

- **Operacionais** – quando aprovados pelo usuário, estão prontos para utilização (geração de produtos acabados. Exemplo: access).
- **Semioperacionais** – quando são necessárias poucas modificações no protótipo.
- **não operacionais** – Quando são voltados para análise de pontos específicos.

A sequência de eventos para o paradigma de prototipação é ilustrado na figura a seguir.

Figura 3 – Ciclo de vida da prototipação

Como todas as abordagens ao desenvolvimento de software, a prototipação inicia-se com a coleta dos requisitos. O desenvolvedor e o cliente reúnem-se e definem os objetivos globais para o software, identificam as exigências conhecidas e

esboçam as áreas em que uma definição adicional é obrigatória. Ocorre, então, a elaboração de um "projeto rápido". O projeto rápido concentra-se na representação daqueles aspectos do software que serão visíveis ao usuário (isto é, abordagens de entrada e formatos de saída). O projeto rápido leva à construção de um protótipo que é avaliado pelo cliente/usuário e é usado para refinar os requisitos para o software a ser desenvolvido. Um processo de interação ocorre quando é feita uma "sintonia fina" do protótipo para satisfazer às necessidades do cliente, capacitando, ao mesmo tempo, o desenvolvedor a compreender melhor aquilo que precisa ser feito.

Modelo incremental

O modelo incremental pode combinar elementos do modelo cascata (aplicado repetidamente) com a proposta iterativa da prototipação. O objetivo principal é trabalhar com o usuário para identificar corretamente os requisitos necessários, de maneira incremental, até que o produto final esteja consolidado.

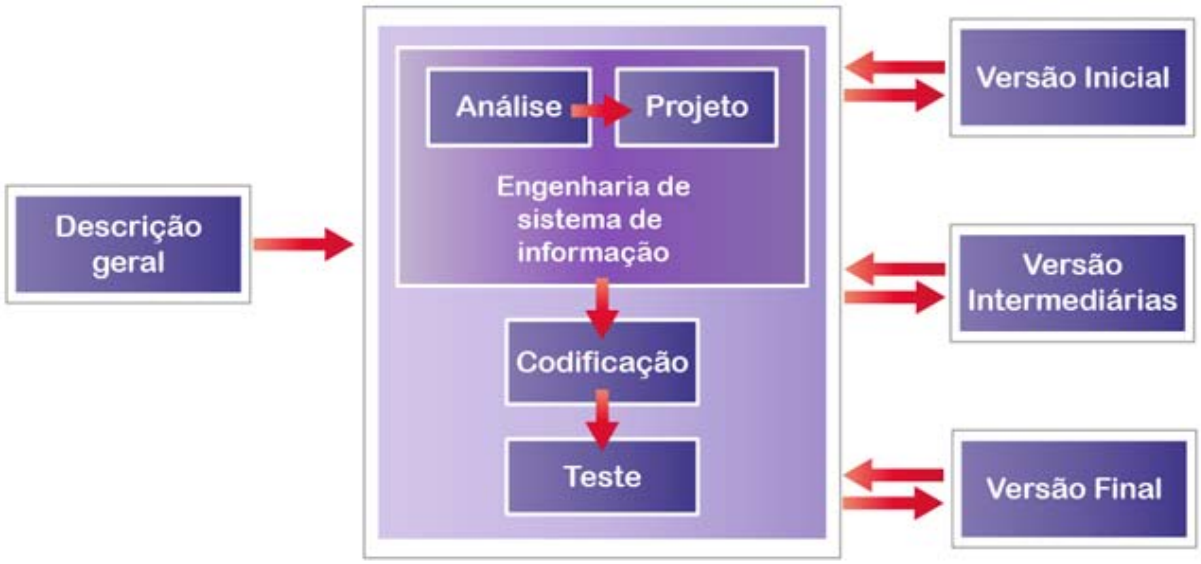


Figura 4 – Modelo incremental

A versão inicial, geralmente, é onde se encontram as principais regras de negócio, que são a essência do sistema. O desenvolvimento é iniciado pelas partes

do produto que estão entendidas e a evolução ocorre quando novas características são adicionadas, de acordo com as necessidades do cliente.

O desenvolvimento incremental é especialmente indicado quando é difícil estabelecer de imediato uma especificação detalhada dos requisitos, sendo indicado para sistemas pequenos, em que os problemas de mudança podem ser contornados.

REFERÊNCIAS

DEMARCO, T. *Análise estruturada e especificação de sistema*. Rio de Janeiro: Campus, 1989.

GANE, Chris e SARSON, Trish. *Análise estruturada de sistemas*. Rio de Janeiro: LTC, 1983.

GANE, Chris. *Desenvolvimento rápido de sistemas*. Rio de Janeiro: LTC, 1989.

AMARAL FILHO, Antônio Rubens Anciães. *Projeto estruturado*: fundamentos e técnicas. Rio de Janeiro: LTC, 1988.

SOMMERVILLE, Ian. *Engenharia de Software*. 8. ed. São Paulo: Addison Wesley.

PRESSMAN, ROGER. *Engenharia de software*. 6.ed. São Paulo: Mc-Graw Hill.