Engenharia de Software

Paulo Cézar Stadzisz

stadzisz@utfpr.edu.br

sala: LIT – Bloco D – 2º andar

Telefone: 3310-4764

Interfone: 4767

Conteúdo:

T1: Introdução à Engenharia de Software (3 semanas – 12 horas)

T2: Processos de desenvolvimento de software (7 semanas – 28 horas) – 1ª parcial = 50%

T3: Especificação de requisitos (8 semanas – 32 horas) – 2ª parcial = 50%

PPGCA Teste de software

Aula 1 – 08/08/16

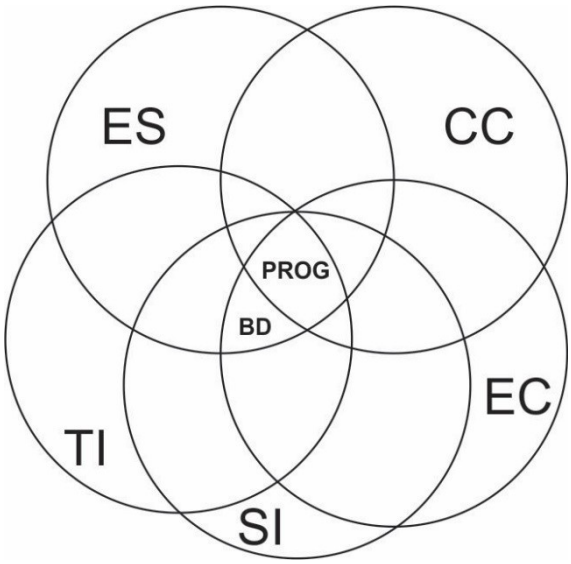
Engenharia de Software

É a área de conhecimento que envolve o emprego de métodos, processos, técnicas, conceitos (teorias), modelos (padrões), linguagens (notações) e ferramentas para o desenvolvimento sistemático de software.

ACM / IEEE

- Computação / computing
  - Grande área temática
  - Áreas de conhecimento (Computer Engineering)

- Engenharia de software
- Engenheiro(a) de software
- Engenharia de computação
- Ciência da computação (Cientista de Computação)
- Tecnologia de Informação (Information Technology)
- Sistemas de Informação (Information System)



Aula 2 – 15/08/16

Qual é a importância de Software (Para a vida)?
Vital

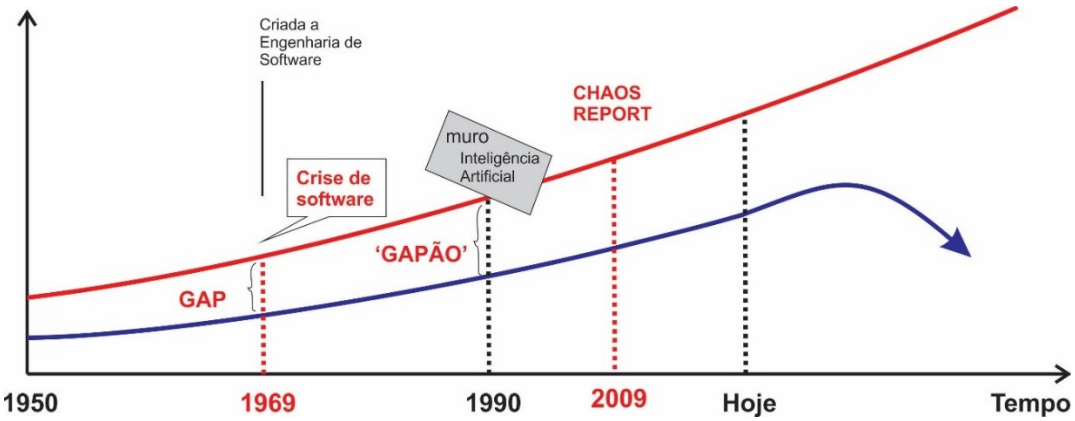
ITAM (Inf. Technology Asset Management) -> IAITAM
- ISO/IEC 19770
- Oxley-Sarbone (Criaram as leis de Governança corporativa).
  - Governança de T.I. (Uma das medidas)

(incentivos Gov. Federal)
Certics - Programa de certificação para empresas devidamente Brasileiras.
- Softex
  - Exportação
  - MPS-BR (Mlehoria de produção de software) (==CMMI)

ABES – estuda tendências

Problemas de Produção de software

- Baixa produtividade
  - LOC = lines of code
  - É uma atividade intelectual complexa
  - Falta de ferramentas mais eficazes
- Altos custos
  - Devido à baixa produtividade
- Complexidade crescente
  - Reduz a produtividade, mas aumenta o número de locs
- Pouca mão de obra qualificada
- Em média, 50% dos projetos são falidos
  - Estatística mundial



** aumento da demanda linearmente e aumento da capacidade*
** produzimos menos do que a demanda*
** gap só vai aumentando, ou seja, estamos em uma crise de sw.*
** criou a engenharia de sw como uma resposta para a crise de sw.*

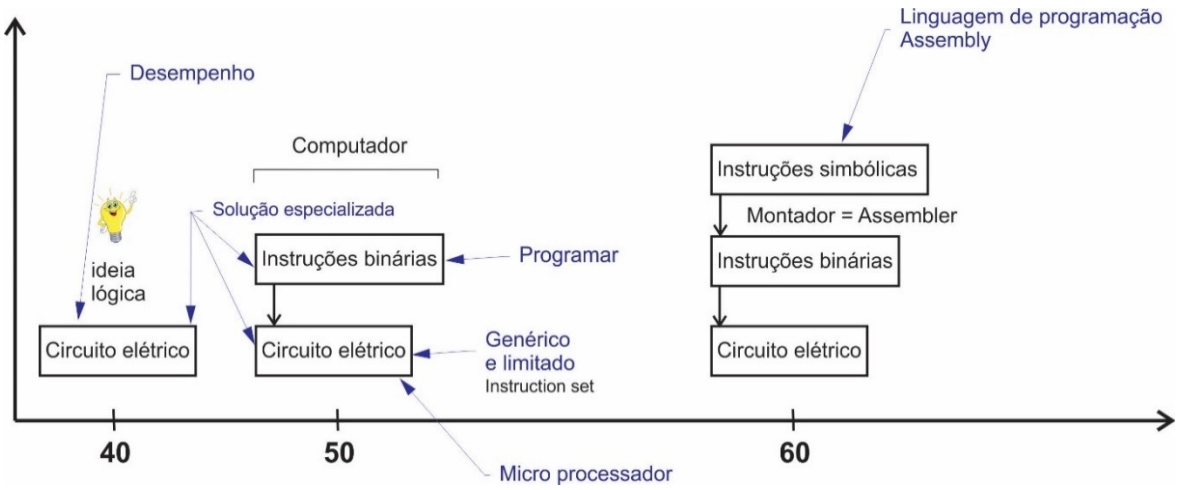
** essa crise contínua que só aumenta o GAP ganhou o nome de "caos".*
** ferramentas erradas para matar a curva vermelha.*
Problema: redução do crescimento populacional.

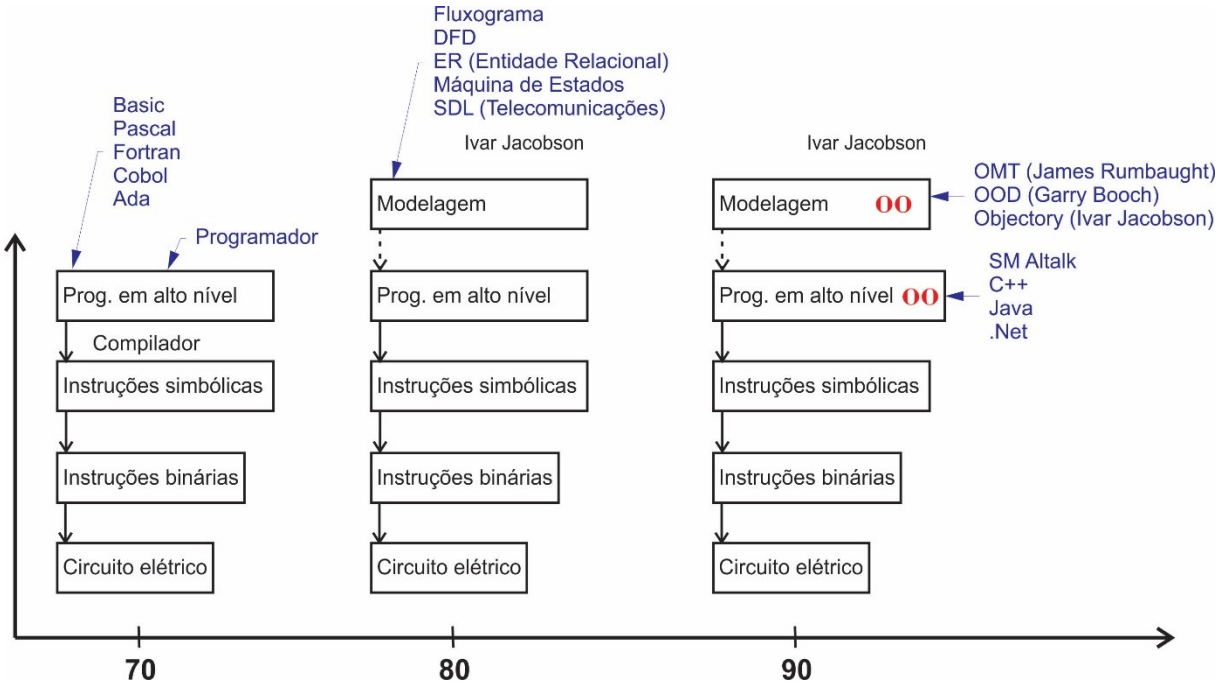
Milagre: aumento macisso da produtividade.

Aula 3 – 22/08/16

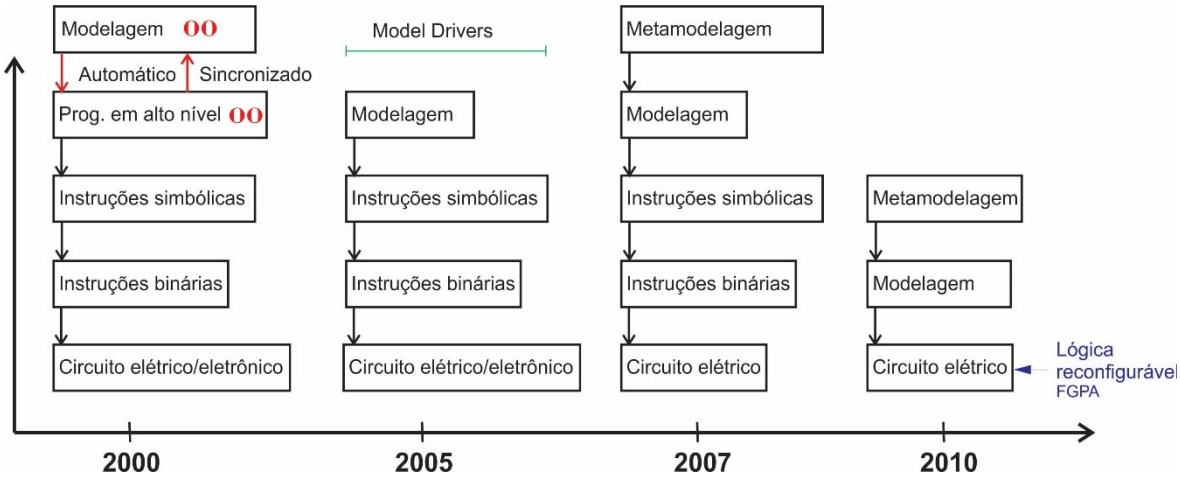
O que é software?

- Arquivos de código fonte (programas);
- Bases ou estruturas de dados;
- Arquivos executáveis;
- Documentação técnica (modelos, especificações, API, ferramentas);
- Capital Intelectual Humano;
- Informações de marketing (clientes, posicionamento, concorrentes);
- Direitos (marcas, registros, patentes) não para direito autoral.

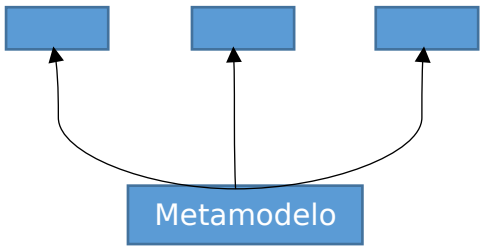




OMT + OOD = Rational UML



Metamodelagem



Server para:

- Reuso
- Especialização
  - Modelagem mais genérica, mais Alto nível.

Software = “macio”, “maleável”, “flexível”.

- Hardware

{ Logicial
{ Logiciaria
{ Logiciel

É a expressão da lógica de funcionamento de um sistema.

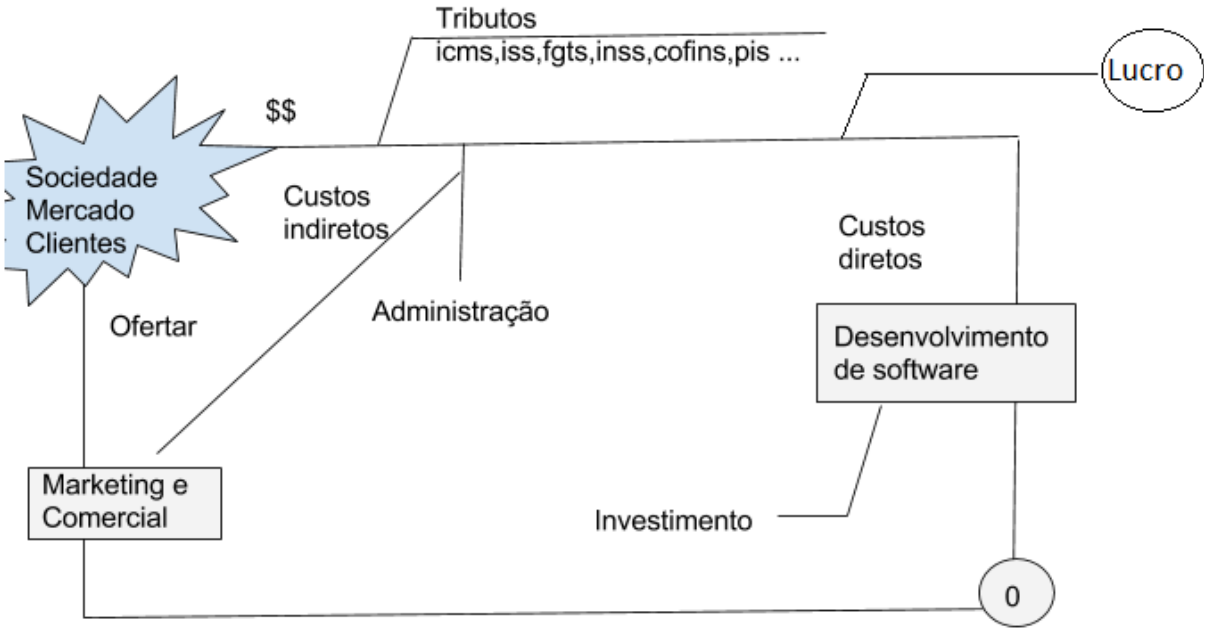
Processos de Desenvolvimento de Software

- Métodos
- Ciclo de vida

Programas de Qualidade

- ISO 9000

Ciclo Econômico de Software



Logicial

-> (Fontes) Investimento:

- Aporte pessoal dos sócios.
- Bancos
- Fomento
- Investidor
- Funding

-> Lucro (Bruto - “Possivel destino”)

- IRPJ
- PLR (Participação de lucros e resultados)
- Reservas R.H, Tributação, Investimento.
- Dividendo aos sócios (50% reserva, 50% Cerveja)

Pontos sensíveis: *Investimento*, *Como seduzir a ‘nuvem’* (Sociedade, mercado, clientes)

Engenharia Engenharia
- Sistema X Software

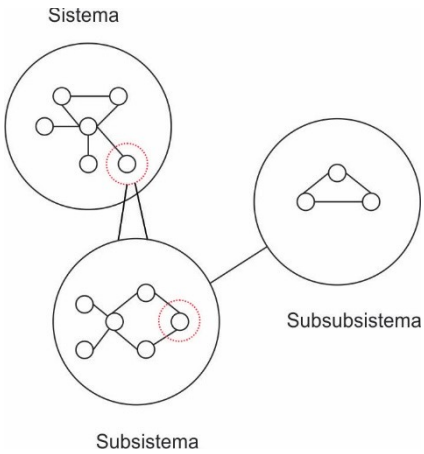
(Sistema: Um conjunto de partes interligadas que cumprem um objetivo específico)
(Sistema: é um conjunto de partes que interagem entre sim com o mesmo objetivo)
System Engineering

Anotações

Método ou Processo

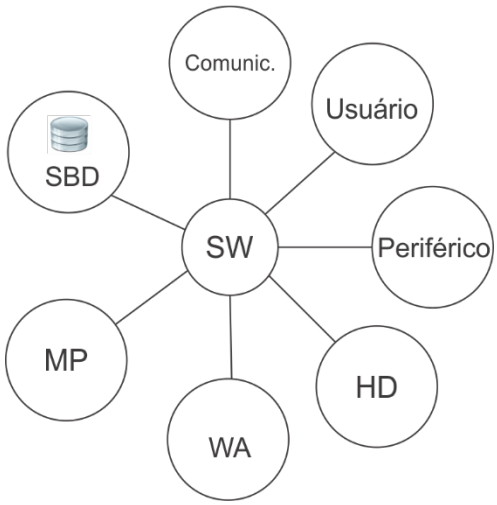
Ciclo de vida = Lifecycle

Sistema: é o conjunto de partes (componentes) que se relacionam (interagem) para atingir um objetivo comum que dá identidade de conjunto.



Top-Down: vai do mais abstrato ao mais

Sistema ≠ Software



Engenharia de Sistemas

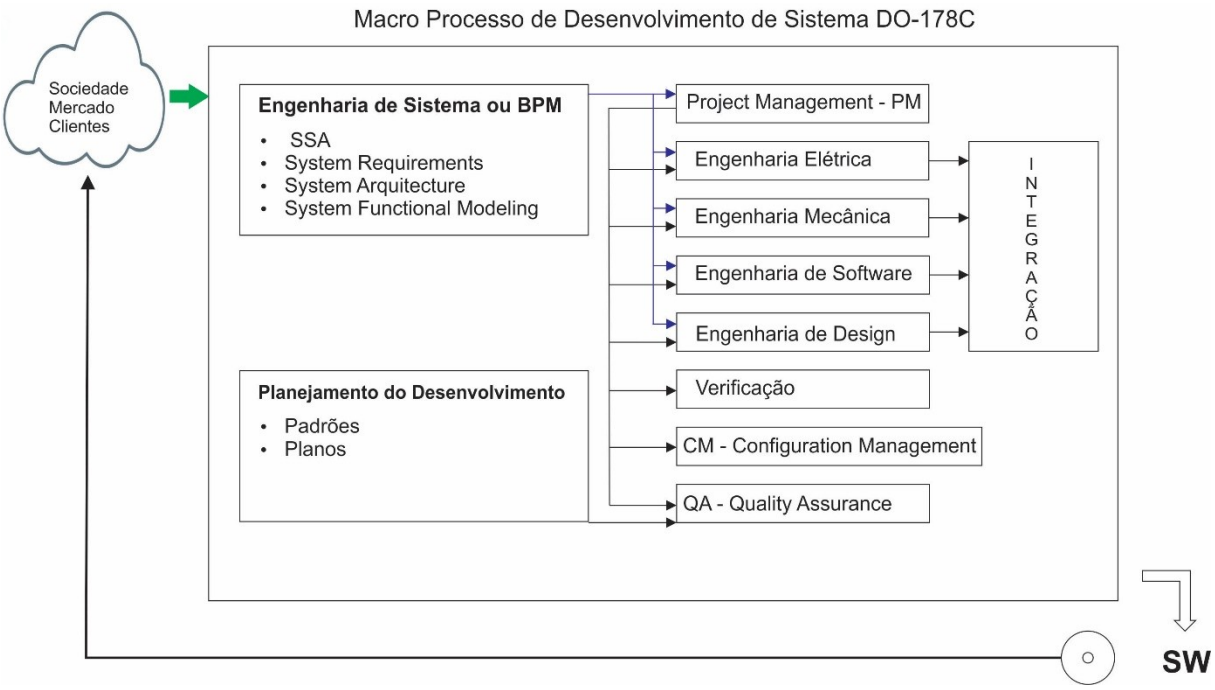
- É a engenharia do conjunto, a qual garante que ao final tudo funcione;
- Modelagem de processos;
- Criada nos EUA em 1940;
- INCOSE – International Council Sistems Engeneering.

SI → Corporativo

BPM – Business Process Modeling é o equivalente a Sistem Engeneering na área corporativa.

Todas as atividades e processos de uma empresa.

“O software serve para ajudar a empresa em alguma etapa do business, para isso a modelagem dos processos é util. Entender os negócios a serem informatizados.”



Project Management – PM

- Administração de empresas
  - Executivo
  - Cronograma físico
  - Cronograma financeiro
  - Milestones
  - Capacitação
  - Alocação de recursos
  - Controle

SSA = System Safety Analysis

- Aeronáutica: Normas – DO – 178C – RTCA
  - A – pifou, morreu
  - B – pifou, machucou (eventualmente morreu)
  - C – pifou, incomodou (eventualmente machucou)
  - D – pifou, notou (eventualmente incomodou)
  - E – pifou, ninguém viu. (seria o sem safety)

IEC EM 50.128

- SW área ferroviária
- SIL – 4
- SIL – 3
- SIL – 2
- SIL – 1
- SIL – 0

FMEA = Failure Modes and Effects Analysis (análise de modos e efeitos de falha)

- modos: como pode ocorrer a falha

- efeitos: resultado da falha

Análise de possíveis falhas numa porta automática

Nº	Modo	Efeito	Classe
1	Falha no sensor de aproximação	Choque mecânico com a porta fechada	C
2	Falha na abertura da porta	Idem	C
3	Falha no sensor de passagem	Imprensar o usuário	C

Partição: quando é possível separar os tipos de falhas no projeto.

System Requirements (requisitos de sistema)

- O que o sistema deverá fazer
- O que o sistema deverá ter como propriedades

Especificação

- Textual
- Gráfica/textual
- Matemática (linguagem Z) formal

System Architecture

- Descrição estrutural / organizacional do sistema e de suas partes.

Exemplo:

Diagrama de classes

Diagrama de componentes (as classes não são mostradas)

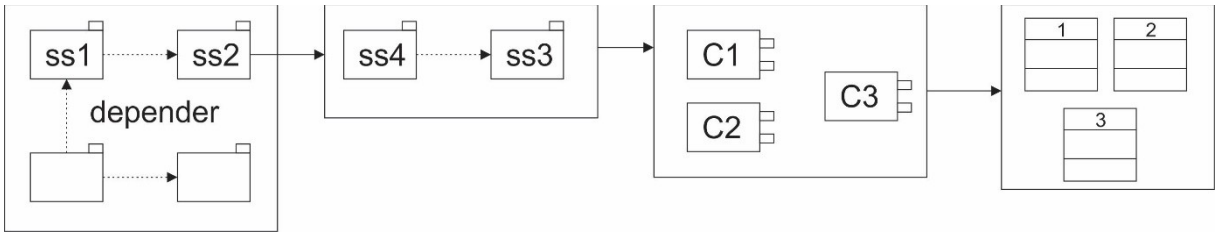
Diagrama de packages (não tem interface, pode colocar várias coisas que não se relacionam).

Arquitetura de SW (UML)

Design

Projeto, concepção { Pensamento
Modelagem ou desenho

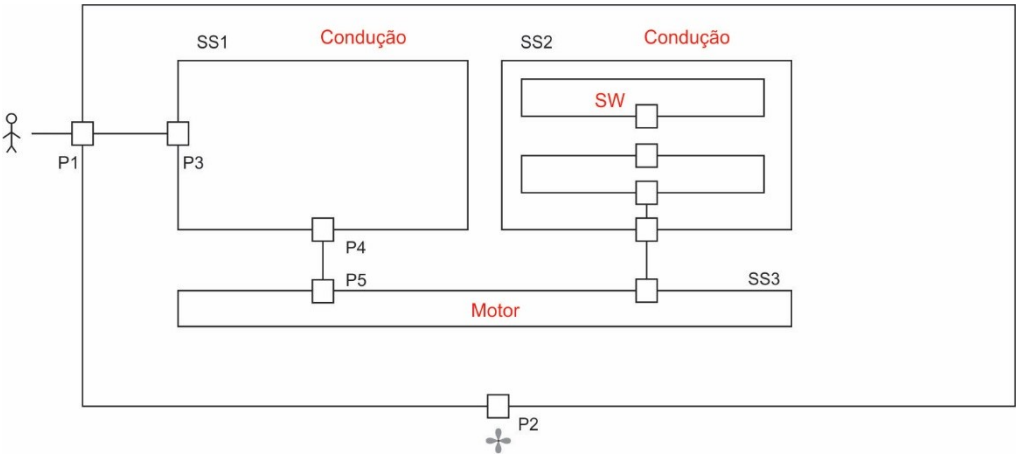
- Diagramas de pacotes



- Colocar junto os casos de uso que têm afinidade;
- Pode-se relacionar os subsistemas;
- Os relacionamentos são dependências;
- Pode-se ver a composição das coisas (mostrando os componentes e as relações)
- Pode-se pegar um componente e detalhar em diagrama de classes.

Arquitetura de Sistemas

- Diagrama de blocos **(SYSML)**



Especificação de Requisitos

- Texto: o sistema deverá permitir ao piloto acelerar o submarino. Ou seja, escrever na forma de texto aquilo que espera-se que o submarino fará.
- Formal: equação (teorema)
- Gráfica: diagrama de requisitos (faz parte da linguagem SYSML), não tem em UML.

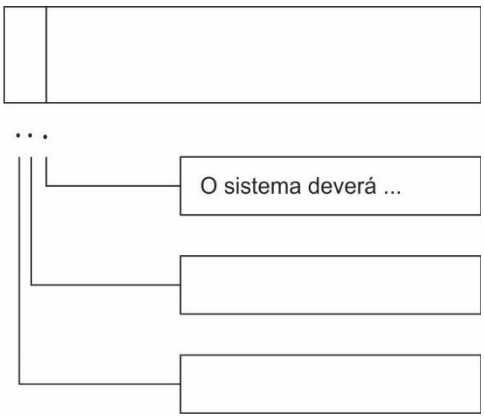
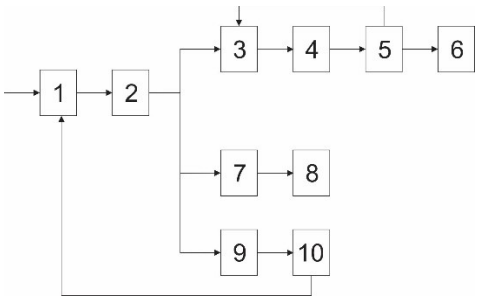


Diagrama de requisitos

Modelagem Funcional de Sistema

- Modelo funcional

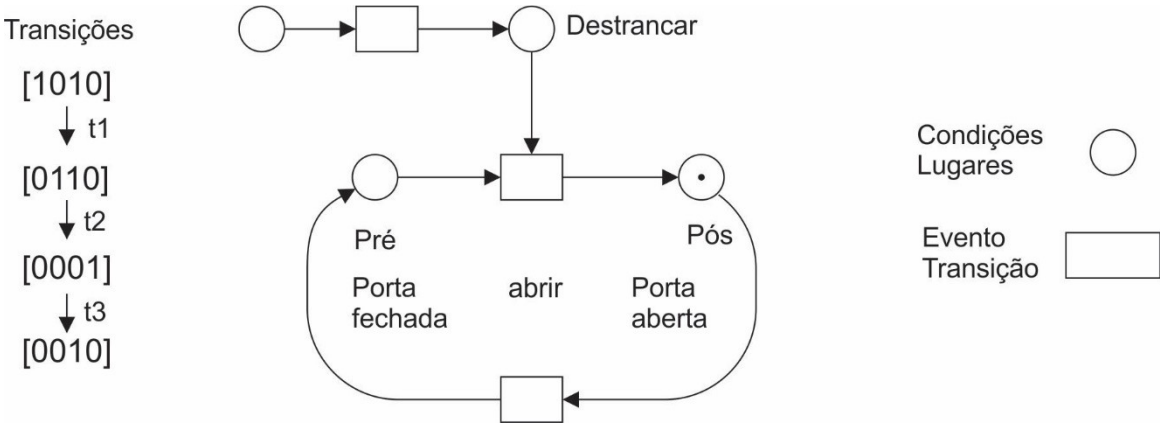


- **Redes de Petri**

Permite modelar sistemas complexos, com muitos elementos acontecendo ao mesmo tempo. Ao contrário da máquina de estados que tem rapidamente uma explosão combinatória de estados. A primeira faz a soma das redes, a segunda multiplica as redes.

- Altamente concorrente
- Linguagem formal (permite análise matemática)
- Não tem explosão combinatória de estados
- Jogável (pode-se analisar a dinâmica dela)
- Representação gráfica
- CPN (ferramenta de linguagem)

Usada com sucesso em engenharia de manufatura.



Máquina de estado

- Diagrama de atividades
- Diagrama de transição de estados

Padrões (Standards)

- Requirements Specification Standard
- Design Standard
- Coding Standard
- Verification Standard

Exercício

Escrever um coding standard

Observações:

1. Escolha uma linguagem de sua preferência
2. Limitado a 5 páginas, sem capa
3. Documento em português
4. Pode copiar da internet

Aula 03/10/16

Exemplos de Planos:

- PSAL – Plan for software aspects of certification
- Software requirements specification plan
- Software development plan (processo de desenvolvimento)
- Software verification plan (teste, revisão, inspeção) -> qualidade técnica
- Software quality assurance plan (plano de garantia de qualidade)
- Software configuration management plan
- Software tools qualification plan (qualificação de ferramentas – compilador, debbuger)

FEE consultant

CM

- Version control
- Branches
- Base Lines -> gerenciamento das entregas
- TASKS
- Change management

} Synergy IBM

Processo de Desenvolvimento de SW

- Organização de atividades para desenvolvimento sistemático de um software.
- Metodologia, método ou ciclo de vida (processo).

Antigos

- Waterfall

- Prototipação
- Modelo em V
- Refinamentos (espiral)

Atuais

- Processo unificado (NUP)
- Métodos ágeis (XP, Scrum, Lean
- Harmony

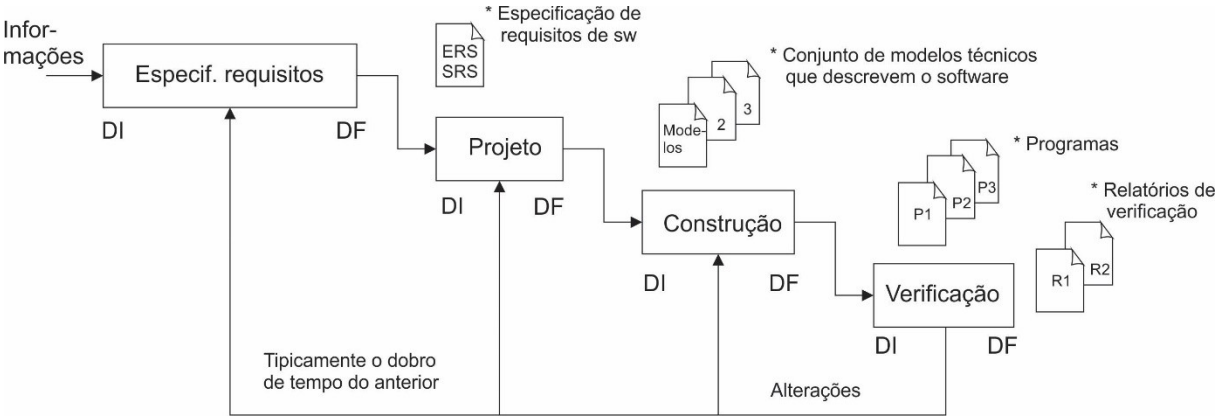
Novas (por vir)

- Model driven
- Axiomatic design

Atividades

- Especificação de requisitos
- Projeto (Design) – inclui a modelagem
- Construção
- Verificação

Modelo Clássico (Waterfall ou cascata)



Waterfall

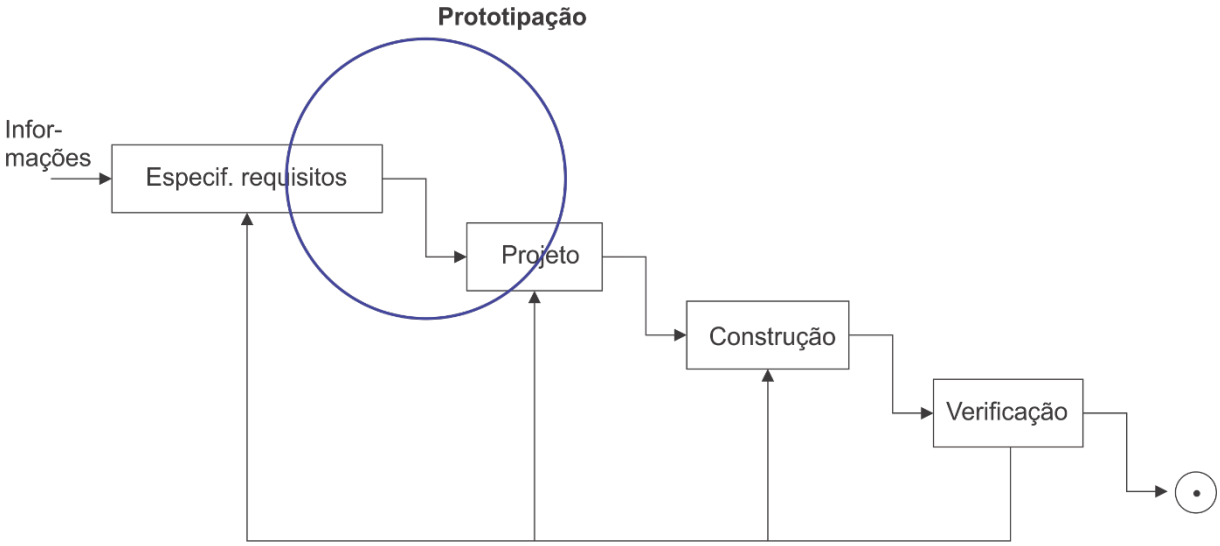
- Sequencial
- Aspectos positivos:
  - Simples e intuitivo
  - Fácil de gerenciar
  - Possível conforto de trabalho
- Aspectos negativos
  - Alto esforço para retornos
  - Alongamento de prazos
  - Alto risco de falhas nos requisitos em razão dos prazos longos

Aula: 10/10/16

Processos de Desenvolvimento de SW

- Processo Clássico
  - Waterfall
    - Sequencial
    - Fácil de entender e gerenciar
    - Alonga o desenvolvimento
      - Deixa muito distante a especificação da verificação
        - Risco de não atender as necessidades do cliente

Protótipos prototipação

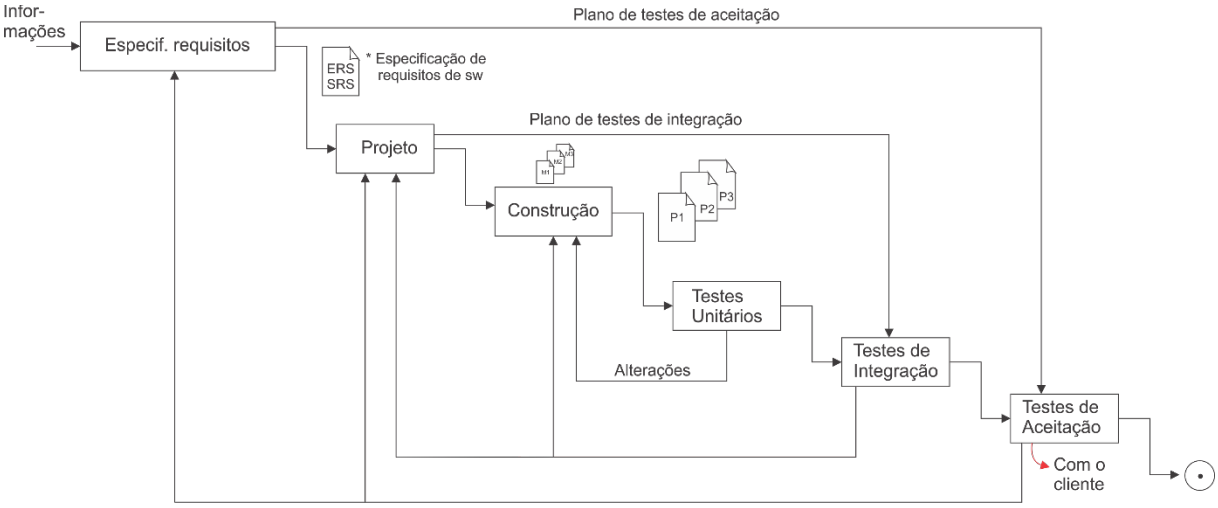


Objetivo: permitir validar (parcialmente) os requisitos.

Regras:

- Investir o mínimo possível na construção do protótipo;
- Jogar fora depois de usar.

Modelo ou Processo em V: “Cascatão”



Int soma (int a, int b)

{

.

.

.

}

caixa preta

caixa branca

Classes de equivalência

Ex: int 1 bite = 1 bit para sinal

7 bits para outros – 127 + 127

Int -127 ... 0 ... 127

-127 **-1**

Casos de teste:

soma (-3, 7)

soma (-5,10)

soma (-5,6)

soma (0,0)

soma (0,9)

soma (5,6)

Coverage Analisys (cobertura de condição)

Modelo sequencial, com destaque para testes.

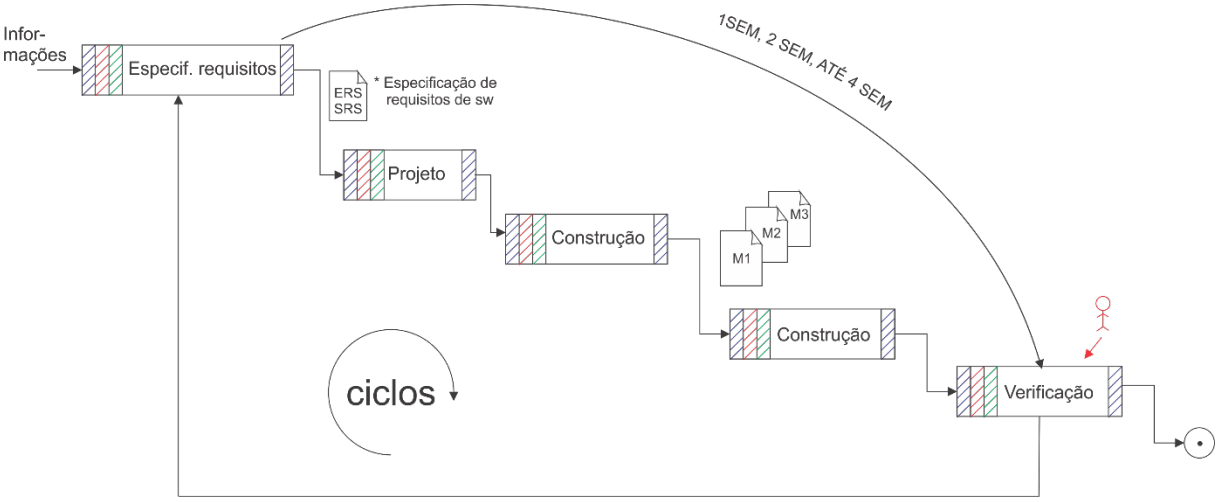
Processos de Desenvolvimento de SW

- Processo por Refinamentos Sucessivos
  (Gane e Sarson) – 1980

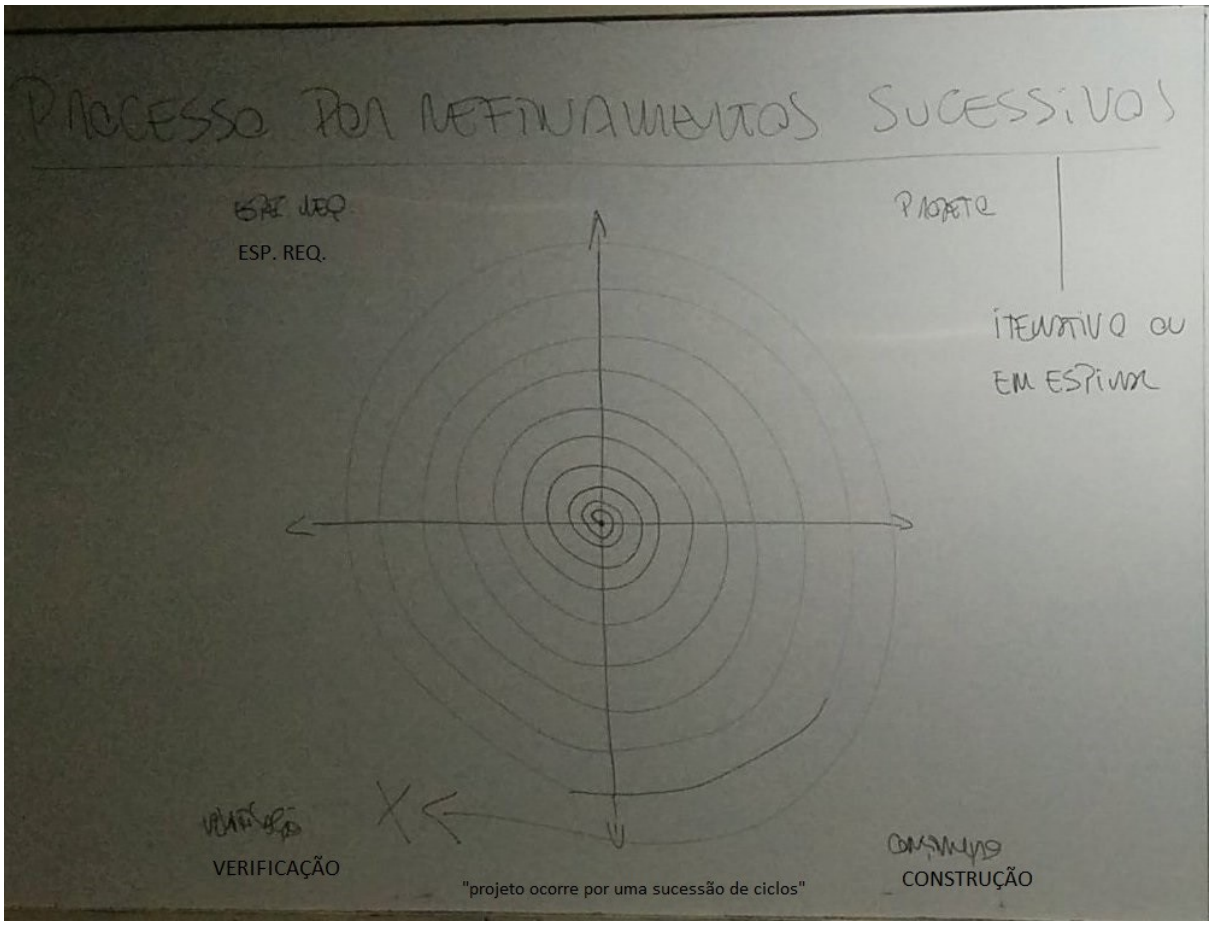
Em lugar de revisar completamente cada atividade antes de passar para a próxima, avança-se um pequeno passo (incremento) de refinamento em todas elas simultaneamente gerando uma nova versão evoluída do SW.

Cada refinamento é chamado ciclo ou iteração.

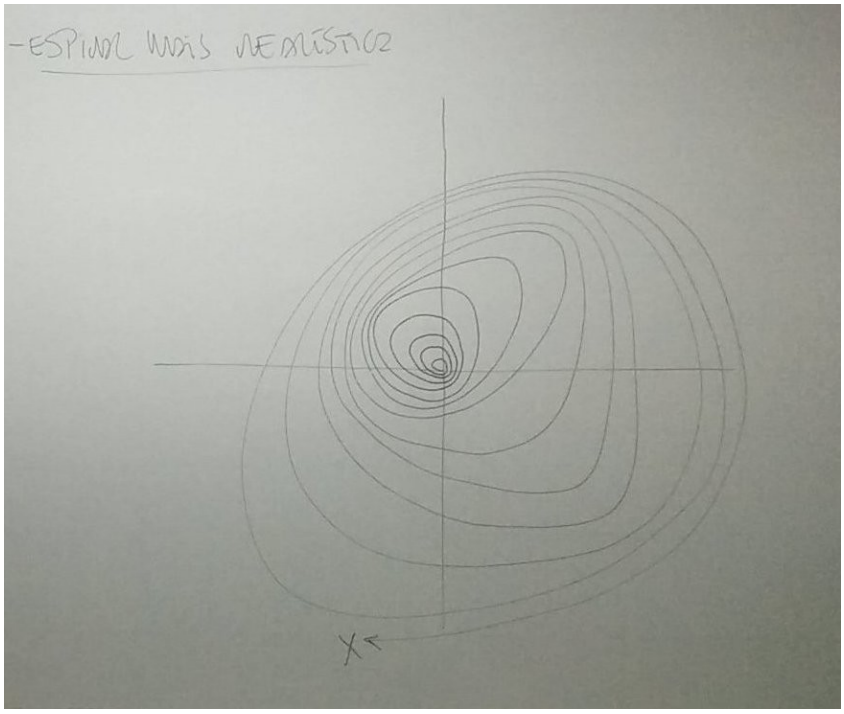
Processo em Espiral



Verificação frequente ajuda a garantir que o produto ao final estará próximo do "desejado" pelo cliente \ mercado.



- Variação nos ciclos (t);
- A carga de trabalho varia em cada ciclo;
- Há variabilidade no esforço durante os ciclos.



Vantagens:

- Reduz o risco e aumenta as chances de sucesso por meio de verificações
- Tendência de ser mais curto do que os processos sequenciais;
- Desenvolvimento em “time” (equipe)
    - **o** Maior corresponsabilidade cooperação sinergia.

Desvantagens:

- Pode gerar dificuldades na equipe heterogênea;
- Dificuldade de entendimento do processo e de sua gestão.

Abordagens para o espiral:

- Abordagem em amplitude;
- Abordagem em profundidade;
- Abordagem mista.

Aula 17/10/16

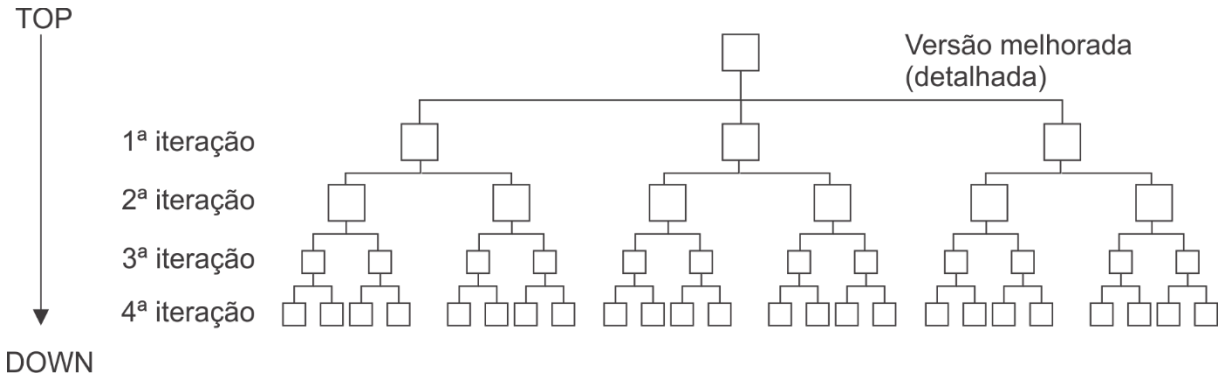
Processo por Melhoramentos Sucessivos:

- Iterativo e incremental, em espiral;
- Organizado na forma de ciclos (ou iterações);
    - A cada iteração revisam-se todas as atividades de desenvolvimento, gerando uma versão melhorada do software (um incremento);
    - uma iteração dura de uma a quatro semanas;
    - revisado por um time (equipe multidisciplinar).

Há duas abordagens principais:

Iterações em amplitude:

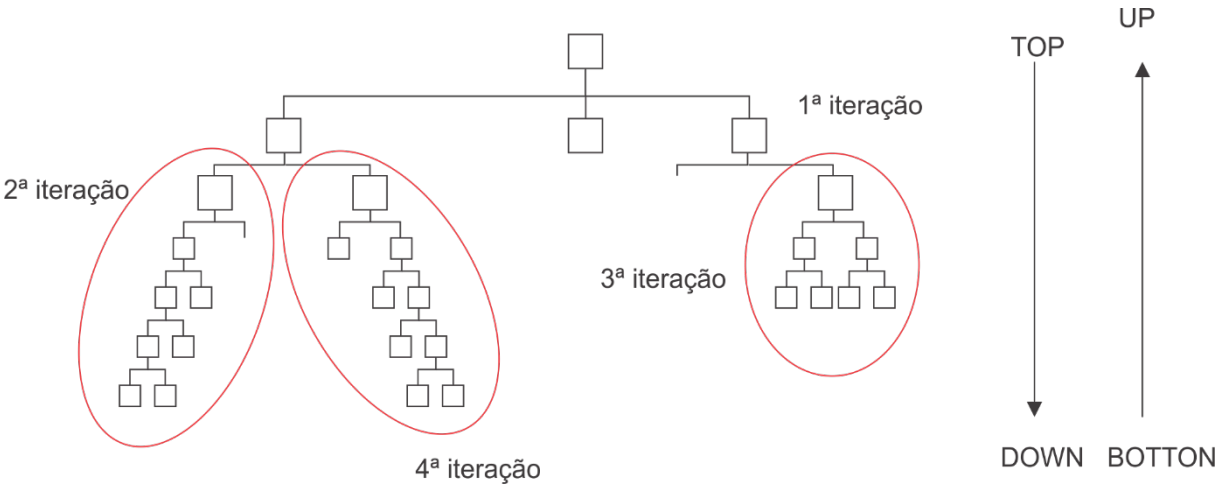
- É o mais adequado para o ponto de vista da engenharia;



*Mais rápido
* Consistentes os módulos

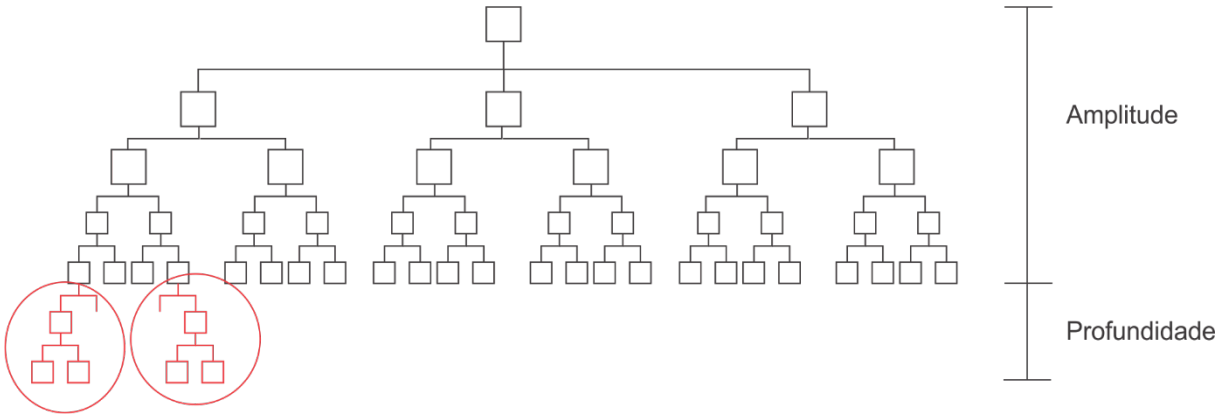
* Arquitetura Intencional (sempre melhor)
* Maior produtividade, sem conflitos, sem remendos
* Vantagem: Coerência (tudo encaixado certo)

Iterações em profundidade:



*Força a barra para encaixar tudo
* pior organizado
* Menor produtividade
* Desvantagem: arquitetura incidental
* Tentativa e erro
* Vantagem: Dinâmico

Abordagem Mista:



Rational Unified Process – RUP

- Criado em 1997 por: Grady Booch - OOD

James Rumbanght – OMT RATIONAL
Ivar Jacobson - OBJECTORY

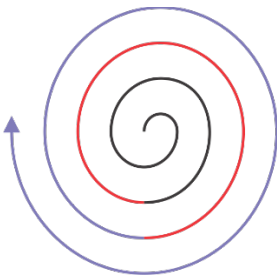
UML – Surgiu da junção das três linguagens já citadas. (gratuito)
RUP - Também surgiu da junção das três linguagens já citadas. (gratuito)
LIVROS
ROSE
REQUISITE PRO

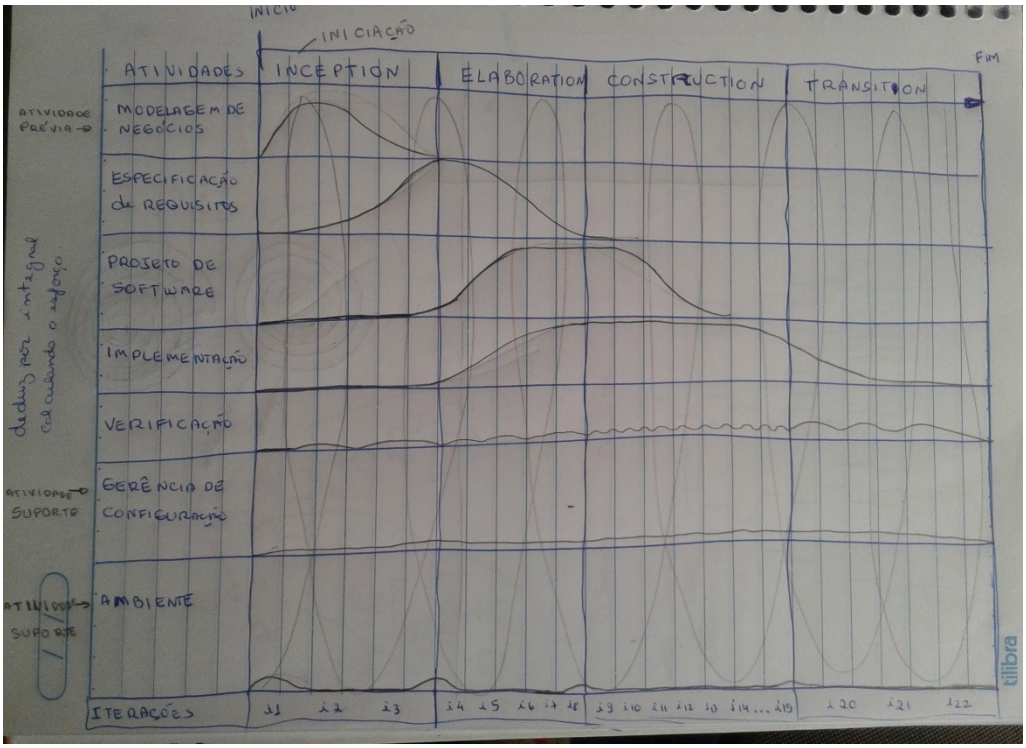
Concorrente da Rational: Telelogic
DOORS
RAPHSODY
SYNERGY

Em 2007 a Rational foi comprada pela IBM.

Em 2009 a Telelogic foi comprada pela IBM.

- Processo mais completo (rico) com relação ao SEBOK (Guide to the Systems Engineering Body of Knowledge);
- SEBOK é um processo iterativo e incremental;
- Inovador: Fases de desenvolvimento
  Curvas de esforço





Métodos Ágeis (Agile Methods)

- Processos Iterativos e incrementais em profundidade.
- Baseados no conceito de “agilidade”.
  - Valores definidos no “Manifesto Ágil”.
    - Indivíduos e interações tem mais valor que métodos e ferramentas;
    - Software funcionando mais que documentação técnica;
    - Colaboração com o cliente mais que contatos;
    - Adaptação a mudanças é mais importante do que planos.

Principais Métodos

XP – Extreme Programming: Melhorar a qualidade e mais dinâmica.
SCRUM
LEAN

Avaliação 1 – 24/10/16

Aula 31/10/6

Episódio III: Especificação de requisitos de sw

- Definição precisa dos objetivos, propósitos ou de "o que" pretende-se alcançar com um desenvolvimento de sw.
- Literatura: > 50% dos projetos falham por não ter especificação de requisitos.
- Os requisitos são provenientes do clientes/mercado. Mas quem especifica é o engenheiro/analista.

PCS: (feeling) 70% SW

90% OUTROS

Qual o problema?

- O cliente não sabe o que quer; (não é verdade)
- O cliente não sabe dizer o que quer; (normal)
- Os requisitos mudam com muita frequência; (não é verdade)
- O engenheiro / analistas pode definir os requisitos baseado na sua existência; (Não)
- Especificação de requisitos superficial, incompleta ou inconsistente. (Não é verdade para nós)

IDEIA:

1. Get Included (envolva-se)
2. Faça uso de técnicas e ferramentas
3. Entender/mapear as razões para os requisitos

Definição de Requisitos

Um requisito é uma exigência (obrigação) imposta pelo cliente sobre o sw a ser desenvolvido.

Tipos (ou categorias) de requisitos:

- **Requisitos Funcionais**
  - Descreve o que o sw deverá fazer (entregar).
  - Exemplo: imprimir o relatório de vendas, cadastrar um livro, gerar um alarme.
- **Requisitos Não Funcionais**
  - Descreve características, propriedades ou restrições
  - Exemplo: ser amarelo, ser escrito em .NET, ser barato, ter botões arredondados.

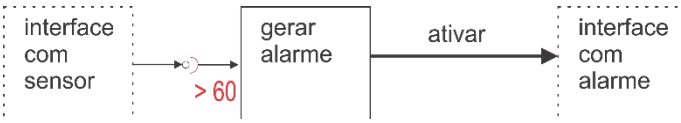
Requisitos
- econômicos: custo, preço, prazo
- desempenho: taxa, tempo, atendimento
- ambiente: SOP, biblioteca, ferramentas, linguagem de modelagem
- estéticas: cor
- estima: marca, referências

Real Time – em tempo real.

- VXWORK
- TOS
- LINUX EMBEDDED
- WINDOWS EMBEDDED
- ANDROID
- CNX
- X-KERNEL (Brasileiro)

Formas de Especificação

1. Textual (criar uma sentença em linguagem natural para cada sentença)
   Ex: o software deverá gerar um alarme quando o sensor detectar temperatura maior do que 60 ºC.



2. Modelos
   - Diagrama de requisitos (SYSML)
   - NOPSRS
3. Métodos Formais

Aula 28/11/16

Exercício – correção em sala

Requisitos Funcionais

RF01 - O software deverá mostrar cada caracter lido do teclado.

RF01.1 – O software deverá ler caracteres do teclado.

RF01.2 – O software deverá mostrar cada caracter lido do teclado na posição atual do cursor de edição no monitor.

RF03 – O software deverá carregar em memória um texto de um arquivo armazenado em um sistema de arquivos e indicado pelo usuário.

RF04 – O software deverá permitir ao usuário gravar o texto editado em um sistema de arquivos.

RF05 – O software deverá permitir ao usuário apagar o caracter a esquerda do cursor de edição quando for pressionada a tecla backspace.

RF06 – O software deverá permitir ao usuário alterar a posição do cursor de edição para aquela posição apontada pelo cursor do mouse quando for clicado seu botão esquerdo.

RF07 – O software deverá permitir ao usuário mover o cursor de edição com as teclas direcionais.

RF08 – O software deverá permitir ao usuário copiar o texto selecionado.

RF08.1 – O software deverá permitir ao usuário marcar um texto.

RF08.1.1 - O software deverá permitir ao usuário iniciar a marcação do texto pressionando a tecla shift.

RF08.2 – O software deverá permitir ao usuário comandar a cópia do texto marcado com a combinação ALT+C.

RF08.3 - O software deverá permitir inserir uma cópia do texto marcado na posição atual do cursor de edição com a combinação ALT + V.

Requisitos Não Fucionais

RNF01 - O software deverá ter como padrão a codificação Unicode.

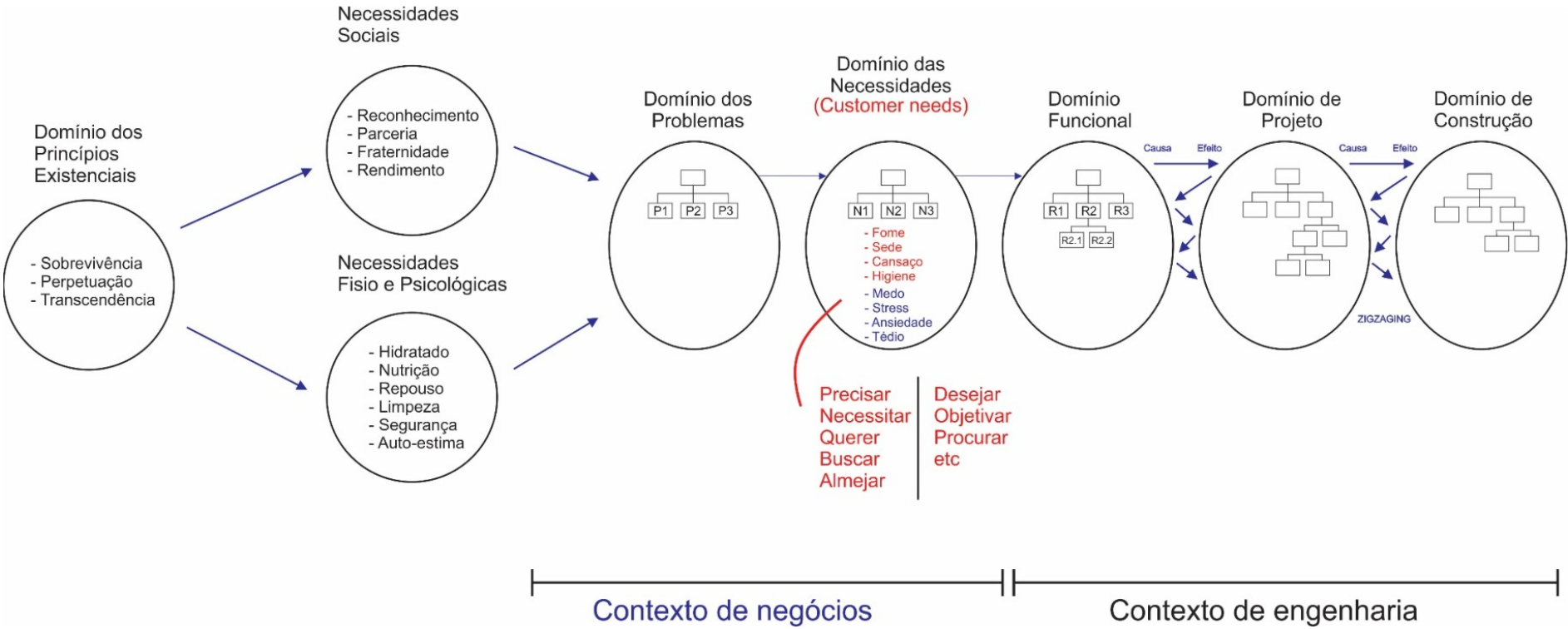
RNF02 – O software deverá ter interfaces com teclado

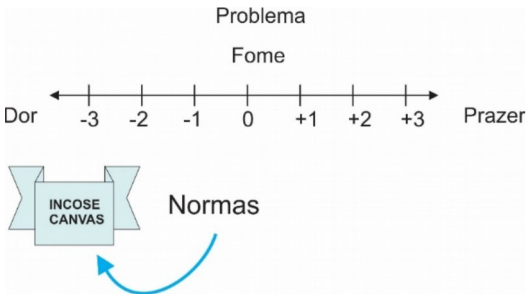
RNF03 - O software deverá ser desenvolvido em até 12 meses.

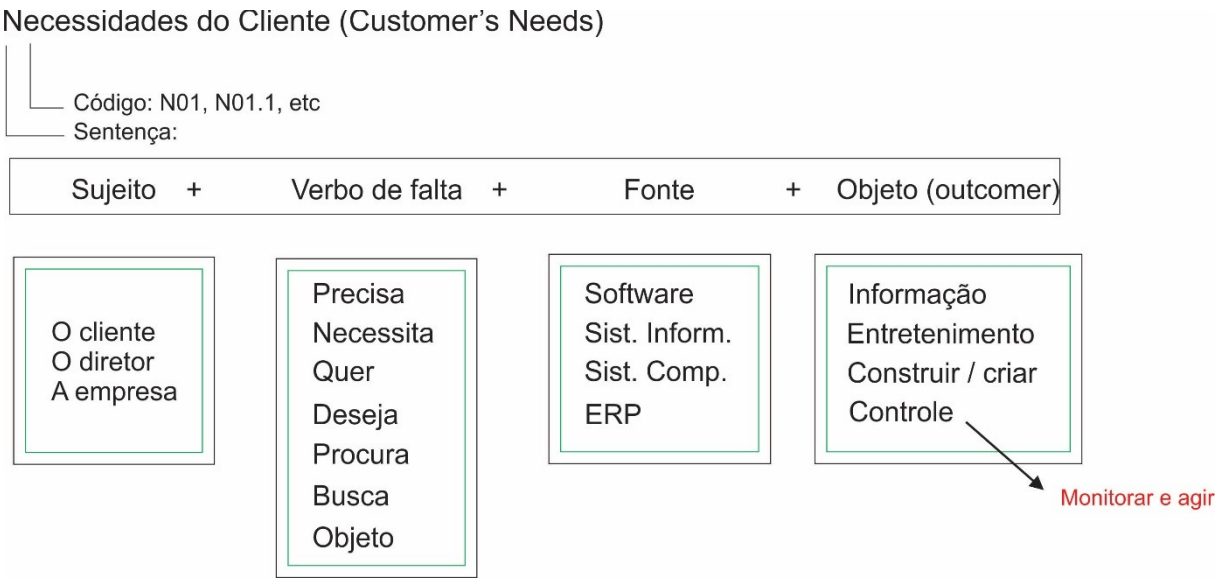
RNF04 - O software deverá permitir funcionar em Windows 10 Linux Manjaro 3.1.

RNF05 - O software deverá ser programado em C++.

RNF06 - O software deverá ter cor de fundo de tela em verde.


Domínio dos Princípios Existenciais
- Sobrevivência
- Perpetuação
- Transcendência
Necessidades Sociais
- Reconhecimento
- Parceria
- Fraternidade
- Rendimento
Necessidades Fisio e Psicológicas
- Hidratado
- Nutrição
- Repouso
- Limpeza
- Segurança
- Auto-estima
Domínio dos Problemas
P1 P2 P3
Domínio das Necessidades (Customer needs)
N1 N2 N3
- Fome
- Sede
- Cansaço
- Higiene
- Medo
- Stress
- Ansiedade
- Tédio
Precisar
Necessitar
Querer
Buscar
Almejar
Desejar
Objetivar
Procurar
etc
Domínio Funcional
R1 R2 R3
R2.1 R2.2
Causa
Efeito
Domínio de Projeto
Causa
Efeito
ZIGZAGING
Domínio de Construção
Contexto de negócios
Contexto de engenharia


Problema
Fome
Dor
-3
-2
-1
0
+1
+2
+3
Prazer
INCOSE CANVAS
Normas



Exemplos de Needs:

O diretor de vendas precisa de um software para saber o total mensal das vendas.

O diretor de RH deseja controlar o horário de entrada e saída dos funcionários.

A assistente de recepção necessita de um software para entreter os visitantes enquanto aguardam.

O médico busca um software para criar as receitas de medicamentos durante uma consulta.

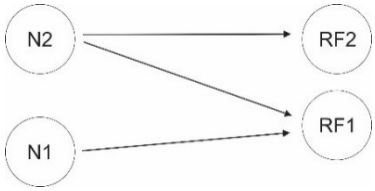
Exercício:

Determine uma ou mais necessidades para um software a partir dos seguintes requisitos:

RF01 – O software deverá autenticar o usuário.

N01 – O cliente precisa de um software para controlar o acesso dos usuários.

N02 – O cliente deseja um sw para saber quais usuários usaram o sistema.

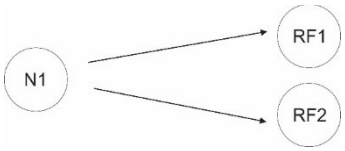


RF02 - O facebook deverá permitir postar uma foto.

N01 – O usuário precisa de um sw para criar posts com imagens.

RF03 – O sw deverá permitir ao usuário cadastrar um novo livro.

N01 – O mercado bibliotecário precisa de um sw para saber quais livros possui.



Necessidades do Cliente (Customer's Needs)

- Código: P01, P01.1, etc
- Sentença:

Sujeito	+	Verbo de intensidade	+	Objeto	+	Penalização
O cliente A empresa O diretor O mercado		deve sofre a expectativa sofre a experança		verbo + complemento		prejuízo dano dor sofrimento ônus

Classes de Intensidade

- Obrigações (deveres)
  - Coisas que devem ser feitas ou sofremos penas.
- Expectativas
  - Coisas não-obrigatórias mas que geram penalização.
- Experanças
  - Coisas que não se aguarda, mas que poderia ocorrer.

Exemplos:

P1 - O diretor de RH deve pagar os funcionários até o último dia útil de cada mês sob a pena de multa sindical e trabalhista.

P2 - A escola deve garantir que todos os alunos saíram antes de fechar, sob pena de negligência na guarda das crianças.

P3 - O diretor de marketing é obrigado a contatar seus clientes semestralmente, sob pena de não cumprir a política de relacionamento da empresa.

Exercícios:

Business Context

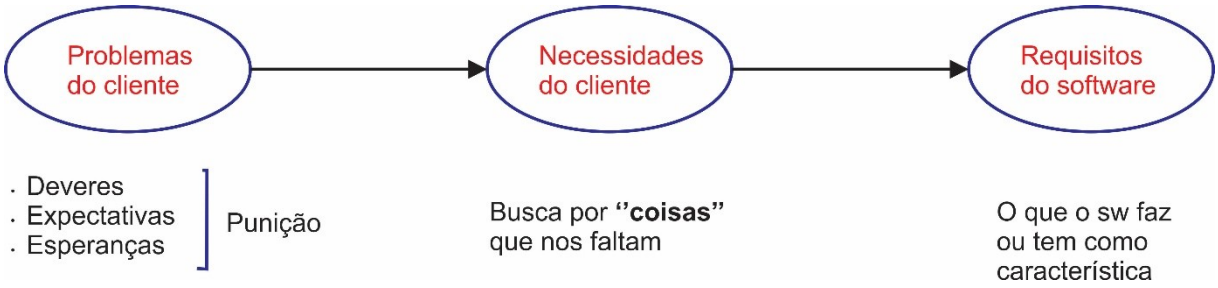
Um diretor de estoque tem preocupações com relação às suas responsabilidades. Determine dois problemas que ele pode ter.

P1 - O diretor de estoque deve garantir as reservas mínimas dos produtos, sob pena de incapacidade de entrega.

P2 – O diretor de estoque deve assegurar que os itens de estoque não sejam danificados, sob pena de perda financeira e incapacidade de entrega.

Aula 12/12/16 – Revisão

Especificação de Requisitos



Exercício:

- Contexto: consultório odontológico
  - Relacionamento e atendimento dos clientes.

Problemas: (obrigações, expectativas, esperanças)

P1 – O dentista deve garantir a consistência de datas/horários no atendimento dos pacientes, sob pena de perdas de receitas e de clientes.

P2 – O dentista deve assegurar que os clientes estão conscientes da data/hora marcada da consulta, sob pena de perdas de consultas.

P3 – O dentista sofre a expectativa dos clientes de serem avisadas antecipadamente sob pena de frustração.

Needs: (o que quero obter do sw)

N01 – O dentista precisa de um sw para controlar o agendamento das consultas.

N02 – O dentista necessita um sw para saber se todos os clientes estão cientes das consultas.

Requisitos funcionais: (o que o sw deverá entregar)

RF1 – O sw deverá permitir registrar um novo agendamento de uma consulta.

RF2 – O sw deverá permitir alterar o agendamento de um cliente.

RF3 – O sw deverá permitir cancelar uma consulta.

RF4 – O sw deverá enviar um e-mail de alerta de consulta aos clientes.