# A Síndrome do *Deadline*: Origem, Causas e Implicações no Processo de Desenvolvimento de Software

**Sildenir Alves Ribeiro[1, 2], Eber Assis Schmitz[1], Antônio Juarez S. M. de Alencar[1], Mônica Ferreira da Silva[1]**

[1]Instituto Tércio Pacitti – Universidade Federal do Rio de Janeiro (UFRJ)
Rio de Janeiro, RJ – Brasil

[2]Coordenação de Automação Industrial – Centro Federal de Educação Tecnológica do Rio de Janeiro (CEFET/RJ) – Rio de Janeiro, RJ - Brasil

sildenir.ribeiro@ppgi.ufrj.br, eber@nce.ufrj.br, ajuarez@nce.ufrj.br, monica@nce.ufrj.br

***Abstract.*** *This paper is the result of ethnological observations in academic experiments applied to the Software Development Process (SDP). Two phenomena commonly applied in project management were studied and involved in the experiment: (1) Student Syndrome and (2) Parkinson's Law. The research was realized in a teaching environment of software engineering with computer science students. The characteristics of the environment and its variables evidenced a third phenomenon, which was denominated "Deadline Syndrome". The experiments also involved the Unified Process to guide the SDP and the Theory of Constraints to identify and treat the restrictive elements found in the production process. As results, the work presents possible causes of the Deadline syndrome and the impacts caused in the PDS and the software product through the qualitative constraints identified.*

***Resumo.*** *Este artigo é resultado de observações etnológicas em experimentos acadêmicos aplicados ao Processo de Desenvolvimento de Software (PDS). Foram estudados e envolvidos no experimento dois fenômenos comumente aplicados em gestão de projetos: (1) a Síndrome do Estudante e (2) a Lei de Parkinson. A pesquisa foi realizada em ambiente de ensino de engenharia de software com alunos de ciência da computação. As características do ambiente e suas variáveis evidenciou um terceiro fenômeno, o qual foi denominado de "Síndrome do Deadline". Os experimentos envolveram ainda o Processo Unificado para guiar o PDS e a Teoria das Restrições para identificar e tratar os elementos restrições encontradas no processo produtivo. Como resultados o trabalho apresenta possíveis causas da síndrome do Deadline e os impactos provocados no PDS e no produto de software através das restrições qualitativas identificadas.*

## 1. Introdução

A produção de software diferencia da produção manufatureira principalmente pelo alto dinamismo que envolve o ambiente e o processo de produção de software. Nos ambientes fabris, geralmente tem-se um processo linear, estruturado e rígido e na maioria dos casos toda a produção é feita por máquinas. Estas máquinas são preparadas e programadas para realizar uma única tarefa, atendendo assim um estágio específico do processo.

O ambiente de desenvolvimento de software é diferente! É dinâmico! E por mais estruturado que seja o processo, não possui tanta rigidez e as perturbações internas e externas exigem que o

processo seja altamente adaptativo. Além disso, todos os artefatos são produzidos por pessoas em todos os estágios do processo, estabelecendo assim o dinamismo do ambiente e do processo de produção de software.

Este trabalho é parte de um estudo maior para identificar gargalos em processo de software. Aqui, serão apresentadas somente as implicações no ambiente e no processo, causados principalmente pelas influências etnográficas. Não é objetivo deste trabalho abordar aspectos comportamentais ou psicológicos dos indivíduos que fizeram parte dos times de desenvolvimento, e sim apontar os elementos restritivos causados pelo fenômeno observado e que influenciaram na máxima produção do software nos três casos experimentados.

O trabalho foi realizado em ambiente de ensino de engenharia de software com alunos de ciência da computação. O Processo Unificado (PU)[1] foi usado para guiar o processo de desenvolvimento e auxiliar na construção dos principais artefatos produzidos. Os princípios da Teoria das Restrições ou TOC em seu acrônimo em inglês para *Theory of Constraints*, foram adotados para atuar sobre o processo na tentativa de identificar e tratar os gargalos qualitativos do PDS. A pesquisa executou 3 rodadas experimentais, e a coleta de dados foi feita em tempo de execução e ao final de cada fase do PU em cada experimento. Posteriormente os dados foram analisados e avaliados e correlacionados entre os três experimentos. Com isso foi possível evidenciar um novo fenômeno, o qual foi denominado de "Síndrome do *Deadline*".

Para melhor caracterizar esta evidência e apurar se o achado apresenta alguma conotação nova, foi feita uma comparação entre outros dois fenômenos largamente difundidos, estudados e aplicados na gestão de projetos: (1) A "Síndrome do Estudante": proposta originalmente por [Goldratt, 1987]; e (2) a "Lei de Parkinson", proposta e apresentada por [Parkinson, 1955] e [Murray e Parkinson, 1958].

A observação da Síndrome do *Deadline* só foi possível devido à prática experimental aplicada a Engenharia de Software. O que mostra quão importante é realizar experimentos em busca de evidências dentro desta vasta área de pesquisa.

## 2. Fundamentos Teóricos: Métodos e Técnicas Usados na Investigação

Um processo de investigação deve ser iniciado através da identificação de uma oportunidade de pesquisa. A relevância dessa oportunidade deve ser amparada e suportada pela revisão da literatura relacionada com o que se deseja investigar, pois é a revisão que dará sustentação ao pesquisador durante a jornada investigativa através da aquisição e construção do conhecimento necessário para o desenrolar da pesquisa.

Este trabalho tem origem em uma extensa revisão da literatura que envolve a aplicação da TOC no Processo de Desenvolvimento de Software, como pode ser observado em [Ribeiro, 2017] e [Ribeiro et al., 2017]. Por conveniência e objetivos específicos, não serão apresentados neste texto os resultados desta revisão. Os trabalhos identificados durante a revisão que possuem alguma correlação com este artigo serão devidamente citados no decorrer das sessões. Portanto esta sessão irá focar na multidisciplinaridade que envolve a pesquisa apontando os métodos e técnicas utilizados neste estudo. Será apresentado ainda nesta sessão um esboço do modelo experimental utilizado, o qual permitiu evidenciar e caracterizar a síndrome do *deadline*.

### 2.1 Teoria das Restrições

A Teoria das Restrições é a base metodológica adotada pra executar o experimento. A TOC pode ser vista como um tipo de metodologia de melhoria continuada, que evoluiu e expandiu sua base metodológica ao longo do tempo [Kim *et al.*, 2008].

---

[1] *O PU foi moldado de acordo com os interesses desta pesquisa e de acordo com as características do ambiente de desenvolvimento. Em função disso, o PU foi denominado de PU-LIKE.*

Proposta inicialmente por [Goldratt e Cox, 2003/06], esta filosofia de gerenciamento produtivo incorpora um aspecto prático na tomada de decisão dentro do ambiente produtivo, baseando-se no princípio de que qualquer limitador de um sistema sinaliza uma restrição do mesmo. Analogamente a uma corrente, a restrição seria o elo mais fraco ou "o menino gordo" conforme a analogia feita por [Goldratt e Cox, 2003/06] em seu livro, "A Meta." Devido às flutuações estatísticas que cada processo ou sistema sofre, a Teoria das Restrições afirma que todo sistema possui pelo menos uma restrição [Goldratt e Cox, 2003/06], [Goldratt, 1997] e [Gupta, *et al.* 2012]. Ao observar um processo ou um ambiente produtivo, esta afirmativa pode ser facilmente comprovada devido a todas as influências externas que o ambiente e o sistema está sujeito.

De forma geral, a TOC baseia-se em cinco passos básicos para fundamentar o processo de melhoria contínua, caso as metas do sistema já tenham sido estruturadas e estabelecidas [Gupta *et al.* 2012]. Os cincos passos da TOC apresentado por [Goldratt e Cox, 2003/06] e [Goldratt, 1997], são:

**Passo 1:** IDENTIFICAR as restrições do sistema que possam impedir a maximização do ganho;
**Passo 2:** EXPLORAR as restrições através de um processo decisório para obter a máxima capacidade da restrição, sabendo que as restrições governam a produção;
**Passo 3:** SUBORDINAR todo o processo produtivo às restrições, inclusive a decisão do passo 2 acima;
**Passo 4:** ELEVAR as restrições, aumentando a performance do sistema de acordo com a capacidade máxima da restrição;
**Passo 5:** RETORNAR ao passo 1, caso uma restrição for quebrada/tratada após a execução dos passos anteriores, para que a inércia não crie uma nova restrição no sistema produtivo.

Os passos acima descrevem o algoritmo da TOC e estabelecem um ciclo iterativo no processo de melhoria continua. Através da inserção dos princípios da TOC no PDS, foi possível identificar e tratar as restrições qualitativas encontradas.

## 2.2 Processo Unificado

O Processo Unificado (PU) foi utilizado na pesquisa para guiar o processo de desenvolvimento de software e alinhar todas as equipes com os objetivos do projeto, com os recursos disponíveis e com os prazos adotados para construção e entrega dos artefatos produzidos.

O PU sofreu algumas adaptações para atender aos interesses desta pesquisa, mas sem que sua estrutura fosse corrompida. Em função disso o PU foi denominado de "*PU-Like*". A figura 1 apresenta um modelo BPMN sumarizado do *PU-Like*.

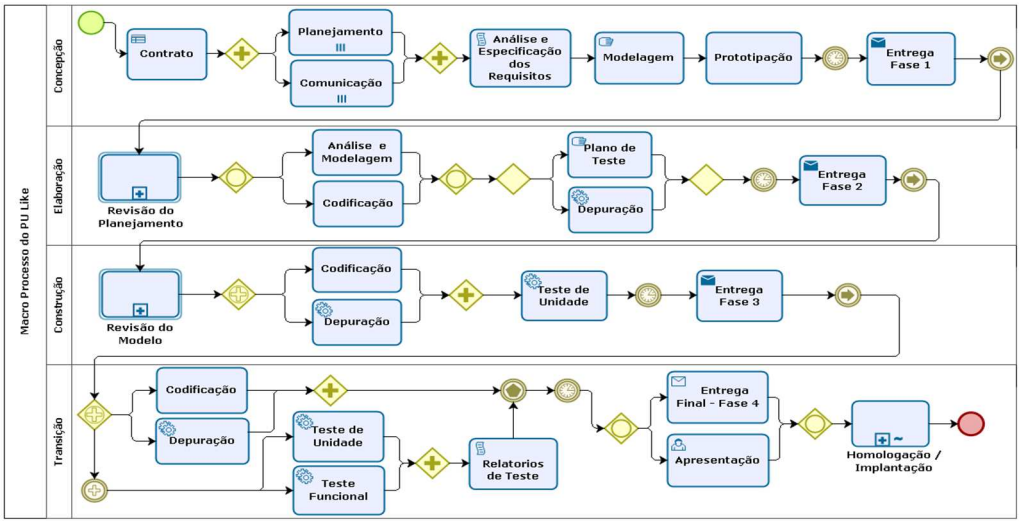


**Figura 1. Modelo BPMN do PU-*Like***

A adoção do PU foi impulsionada pela sua estrutura evolutiva: o PU é interativo, incremental e adaptativo [Jacobson *et al.*, 1999a], [Jacobson *et al.*, 1999b] e [Larman, 2004]. E também por ser um processo maduro e comumente usado na indústria de software.

### 2.3. Experimentação em Ambiente de Aprendizado em Engenharia de Software

O ambiente de aprendizado em engenharia de software foi escolhido para executar este experimento devido às facilidades de propor um problema específico e trabalhar esta proposta com diferentes equipes. Acredita-se que isso seja uma limitação natural em um ambiente real de desenvolvimento, uma vez que em uma *software house*, dificilmente várias equipes podem trabalhar em um mesmo projeto desde a concepção até a transição, onde cada equipe construiria um mesmo produto, assim como uma linha de produção de uma fábrica. Além disso, o ambiente de ensino de engenharia de software é ideal porque não depende de muitos recursos para executar o experimento, diminuindo assim os riscos associados ao projeto e consequentemente à pesquisa.

O trabalho de [Carver *et al.* 2003] apresenta considerações relevantes envolvendo benefícios para estudantes, pesquisadores e instrutores em pesquisas realizadas com estudantes e em ambiente de ensino. A desvantagem que pudemos enxergar é a qualificação da equipe, o que difere de uma equipe de um ambiente real, que conta com profissionais graduados, capacitados e treinados em determinadas ferramentas, modelos, processo e ambientes de desenvolvimento. Em nosso caso, esta foi à primeira restrição encontrada.

### 2.4. Objetivos do Experimento e Metodologia Aplicada

Inicialmente, a ambição principal e única da pesquisa era responder às seguintes perguntas: (1) Quais são os gargalos do PDS? (2) É possível aplicar a TOC ao PDS para identificar e tratar os gargalos produtivos? Uma vez que assumimos que existem gargalos em todo sistema ou processo, dado a afirmação de [Goldratt e Cox, 2003/06] e [Goldratt, 1997].

Mesmo com o objetivo principal traçado, não foi descartado a possibilidade de o experimento apresentar evidências secundárias, ou que de certa forma não fosse o alvo da investigação. Com isso, foi possível constatar a síndrome do *deadline* e suas influências no processo produtivo de software. Primeiramente o fenômeno foi catalogado e caracterizado como uma simples restrição do sistema. Mas diante das repetições dos experimentos e da continuidade das observações, foi evidenciado que a síndrome é um fenômeno que influência no desempenho das equipes de desenvolvimento criando restrições e gargalos qualitativos no ambiente, uma vez que o fenômeno observado não está associado a um domínio específico ou somente a uma determinada equipe. É um fenômeno que está associado principalmente com o ambiente, o processo e os *stakeholders*.

#### 2.4.1 O método científico

O método científico é um conjunto de técnicas para a investigação de fenômenos, aquisição de novos conhecimentos, e corrigir ou integrar conhecimentos prévios [Goldhaber e Nieto, 2010].

O método científico é importante porque orienta o experimento através de uma sequência de passos projetados de acordo com a base teórica da pesquisa.

Para atender as necessidades específicas deste trabalho, ajustamos o método científico proposto por [Goldhaber e Nieto, 2010] de acordo com as cinco etapas da Teoria das Restrições, sendo possível assim estabelecer um modelo cíclico que pode cobrir todo o processo de desenvolvimento do software, como pode ser visto na figura 2.

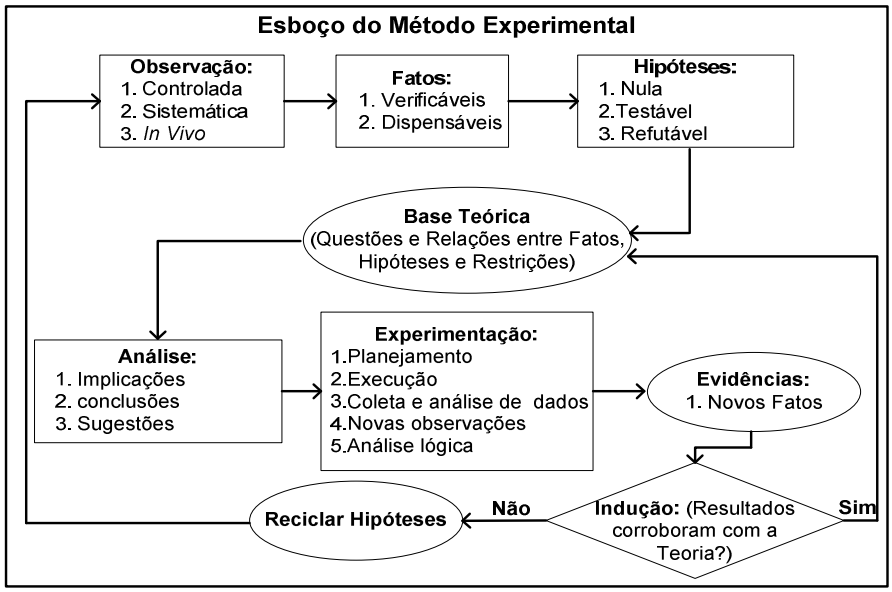


**Figura 2: Método científico aplicado (adaptado de [Goldhaber e Nieto, 2010])**

### 2.4.2 Processo metodológico de execução dos experimentos

A metodologia, os critérios adotados e os métodos de trabalho apoiaram-se em alguns trabalhos disponíveis na literatura que aplicam a TOC para identificar e tratar gargalos no processo produtivo, como: [Sengupta, 2008], [Schragenheim & Dettmer, 2000] e [Pandit, 2009].

É importante colocar que estes trabalhos abordam questões, ambientes e processos específicos para a produção de produtos manufaturados, mas que a partir de uma análise criteriosa pôde-se verificar que o processo e a metodologia dos mesmos podem suportar o ambiente de produção de software. Com isso, foi definido uma estrutura metodológica que estabelece os critérios para a construção e execução do modelo experimental contendo os seguintes tópicos:

- Definição de método de trabalho;
- Desenvolvimento baseado na orientação do *PU-Like*;
- Definição de resultados (artefatos e número de artefato por rodada);
- Definição de recursos a serem utilizados no experimento, inclui: Hardware, Software;
- Criação do livro de registro do experimento;
- Elaboração de critérios para o controle e a evolução do experimento;
- Definição do formato e dos requisitos mínimos de cada artefato a ser entregue.

A definição destes critérios é importante porque define as regras de construção do produto de software e determina os recursos necessários e disponíveis para cada equipe de trabalho. Além disso, permite organizar a pesquisa e mapear o fluxo de trabalho para executar o experimento.

Ainda compõe a metodologia, alguns marcos que foram pré-estabelecido para orientar o desenvolvimento do processo: como:

1. **O ambiente externo:** envolve dentre outras, a fase de estudos exploratórios e a criação dos times de desenvolvimentos que irão representar as máquinas e as linhas de produção no *shop*;
2. **O planejamento do PDS:** baseia-se principalmente no *PU-Like* e nas disciplinas elencadas para rodar o processo além da identificação do caminho crítico e da corrente crítica;

3. **A modelagem do processo:** é a etapa em que é construído um modelo de execução do projeto envolvendo o PDS, a TOC o esquema de escalonamento de tarefas (JSD) e a metodologia de desenvolvimento;
4. **A execução do processo:** que é estabelecida pelas diretrizes de execução do PDS, dado pelo *PU-like* e pelo processo experimental.
5. **O gerenciamento e controle do PDS:** composto pelas atividades para guiar o processo a cerca dos resultados esperados.
6. **Os resultados:** Os resultados são os artefatos produzidos em cada fase do *PU-Like* e que compõe o produto final.

A figura 3 apresenta um esboço dos marcos metodológico para o processo de construção e execução do experimento. A figura mostra ainda os fluxos que orientam a execução do processo e a expectativa mediante a execução de cada marco/ subprocesso.

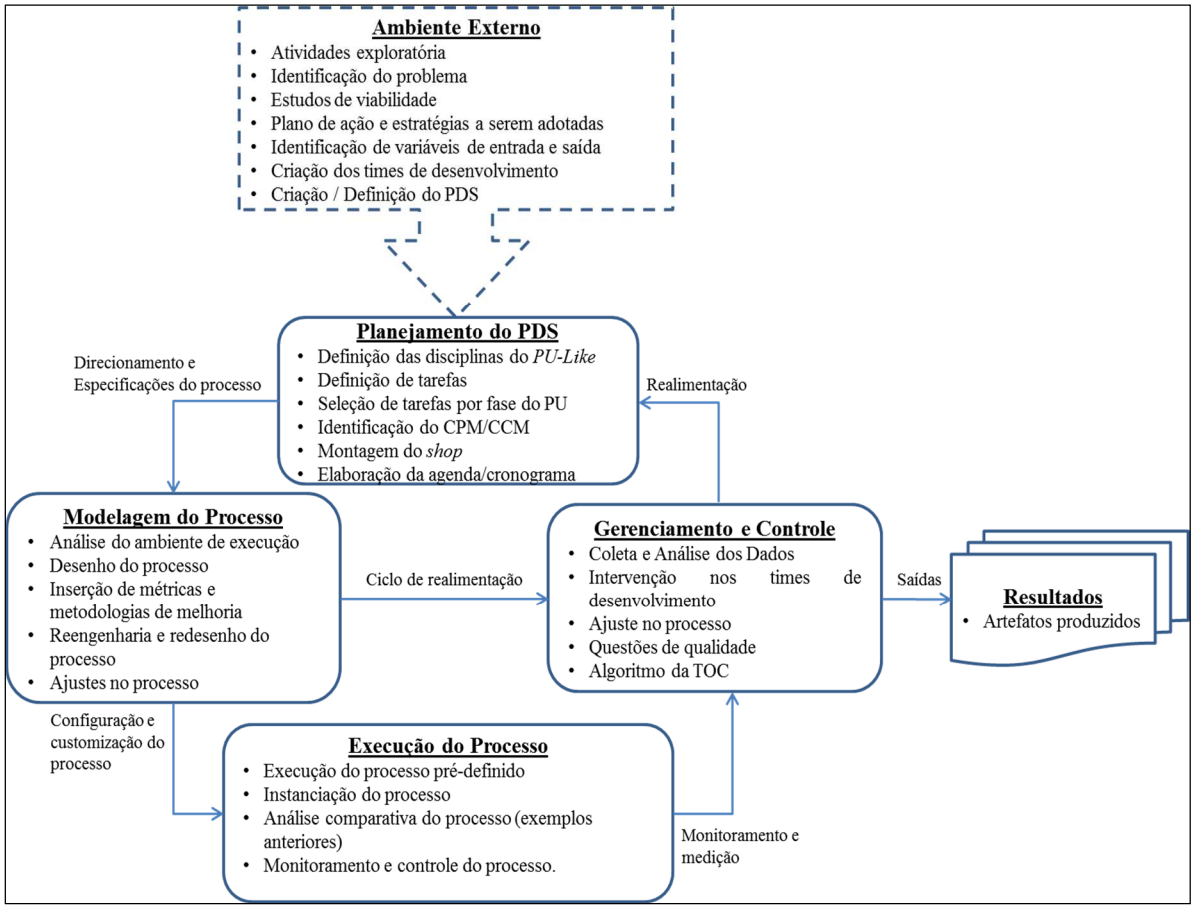


**Figura 3: Esquema metodológico do processo de execução dos experimentos**

### 2.4.3 Organização da pesquisa

A pesquisa foi organizada em dez estágios com as seguintes especificidades:

1. Divisão da classe de forma aleatória, em grupos de trabalho que definem os times/equipes de desenvolvimento;
2. A avaliação inicial das equipes através de questionários para identificar Fatores Críticos de Sucesso (FCS), dado pela experiência dos participantes abrangendo estritamente os fatores

de desenvolvimento de software. Os questionários foram aplicados em dois momentos distintos em cada experimento.

3. Apresentação da proposta (domínio do problema) e consequentemente do produto a ser desenvolvido.
4. Definição dos recursos a serem utilizados por todas as equipes, que inclui hardware, software e laboratórios com infraestrutura de TI;
5. Definição da metodologia aplicada no experimento e criação do livro de anotações do experimento;
6. Apresentação do cronograma com a definição das datas de entregas dos artefatos produzidos em cada fase do experimento seguindo a orientação do PU;
7. Avaliação das equipes durante as fases de desenvolvimento de acordo com as entregas dos artefatos.
8. Elaboração de critérios de alinhamento dos artefatos entregues por cada equipe em cada fase.
9. Avaliação final realizada após a entrega final do produto, que consiste no produto de software instalado testado e homologado.
10. Homologação do produto em ambiente de produção.

A figura 4 apresenta um processo que define o fluxo de trabalho para a execução do experimento, construído a partir dos estágios acima apresentados.

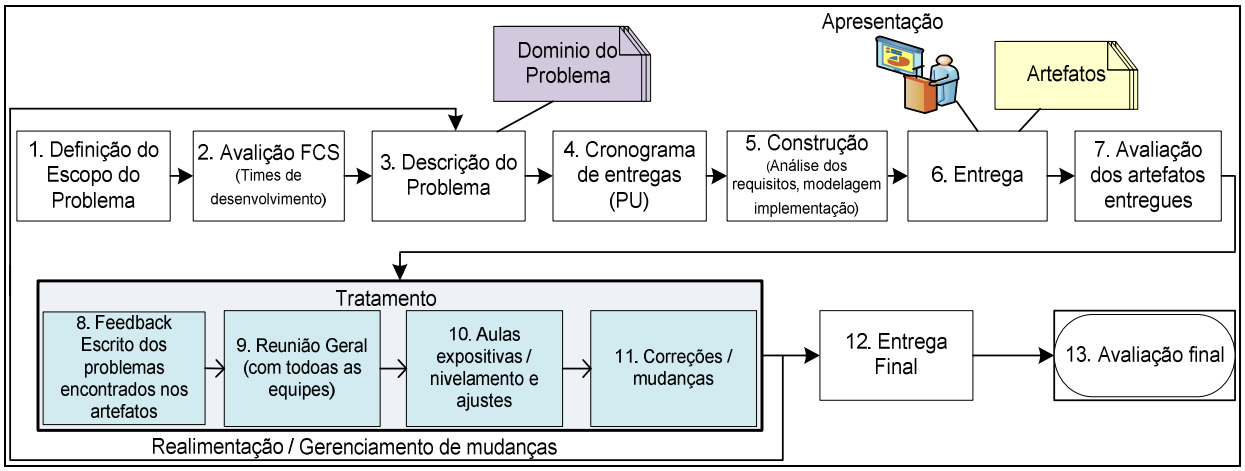


**Figura 4. Fluxo de trabalho para a execução do experimento**

### 2.4.4 Dados Experimentais

Este trabalho foi conduzido de acordo com o plano experimental descrito na sessão 2.4.3, e de acordo com a estrutura metodológica de execução do experimento apresentada e figura 3.

O estudo foi realizado em três rodadas experimentais, denominado de Exp 1, Exp2 e Exp 3. Em cada experimento foi aplicado um domínio especifico, onde:

- Domínio do Problema do Exp 1: Sistema de Controle de Aluguel de Veículos;
- Domínio do Problema do Exp 2: Sistema de Controle de Acadêmico;
- Domínio do Problema do Exp 3: Sistema de Controle de Vendas de Passagens.

A tabela 1 apresenta os dados quantitativos dos três experimentos.

**Tabela 1: Dados quantitativos dos experimentos**

| Dados Experimentais | | | | |
|---|---|---|---|---|
| **Id** | **Variáveis** | **Valores** | | |
| | | **Exp1** | **Exp2** | **Exp3** |
| 1 | Período de realização | 12 /08/2014 – 04/12/2014 | 17 /03/2015 – 16/07/2015 | 13 /10/2015 – 17/03/2016 |
| 2 | Duração em semanas | 17 | 18 | 19 |
| 2 | Número total de alunos participantes | 35 | 32 | 18 |
| 3 | Número de alunos de outras áreas | 03 | 02 | 0 |
| 4 | Número de Linhas de Produção – LP (grupos) | 10 | 09 | 06 |
| 5 | Número de Linhas de Produção defeituosas | 01 | 01 | 02 |
| 5 | Número máximo de máquinas / LP | 04 | 04 | 04 |
| 6 | Número mínimo de máquinas / LP | 02 | 03 | 02 |
| 7 | Número de aulas teóricas | 17 | 18 | 18 |
| 8 | Número de aula práticas | 17 | 18 | 19 |
| 9 | Quantidade de visitas *in-loco* | 4 p/LP | 4 p/LP | 4 p/LP |
| 10 | Quantidade de Sessões de *feedback* | 4 | 4 | 4 |

### 2.4.5 Definição dos entregáveis

Os entregáveis são os artefatos que devem ser construídos, entregues e apresentados em cada fase do *PU-Like*. Cada artefato é originado de uma tarefa escalonada no ambiente de produção. Ao todo, foi definidos um conjunto de 27 tarefas agrupadas da seguinte forma:

- Documentação: Compreende de conjunto de artefatos composto por: (1) relatório técnico descritivo sobre o produto a ser desenvolvido e todos os seus componentes (análise de requisitos, modelagem, código, modelos de base de dados e os testes realizados); (2) modelagem que são artefatos composto pelos seguintes diagramas UML: casos de Uso, classes, sequência e atividades.
- Arquitetura Lógica: Composto de artefatos de modelagem arquitetural e rotinas/código instituídos para modelar e estabelecer a troca de mensagens entre as camadas do sistema e as camadas de aplicação, de acordo com o modelo arquitetural elaborado a partir do MVC (*Model*, *View*, *Controller*).
- Baco de dados: Consiste na construção de um banco de dados My SQL, bem como um conjunto de tabelas relacionais e scripts SQL para armazenar a entrada de dados a partir das especificações das funções do sistema.
- Codificação: Conjunto de artefatos construído em Java de todas as funções do sistema (casos de uso), o que inclui: telas, classes Java, rotinas de teste e executáveis para executar o sistema em *desktop*.
- Teste: Composto de artefatos para execução de teste de unidade e teste funcional. Inclui: Plano de teste, classes de teste e os resultados dos testes aplicados.

## 3. A Síndrome do *Deadline*

A síndrome do *deadline* é uma variação da Síndrome do Estudante e da “Lei de Parkinson”, que foi adotada nesta pesquisa para identificar as pertubações antes de cada entrega em um processo de desenvolvimento de software.

A principal causa deste fenômeno é a priorização de tarefas externas ao ambiente de desenvolvimento, mas também pode ser causado por priorização de tarefas internas, que estão no

escopo do projeto, mas que ainda não foram escanolonadas no ambiente por questões de prioridade de execução ou restrições de recursos.

A priorização destas atividades consomem recursos do projeto, e isto causa impacto direto no processo e no produto a ser desenvolvido. A priorização de tarefas externas ou tarefas não programadas, direciona as pessoas e consequentemente as equipes de desenvolvimento a seguir um fluxo alternativo e com objetivos divergentes do escopo inical, afetando o desenvolvimento e a qualidade dos artefatos, uma vez que o processo passa a ser guiado pela data de encerramento e não pelo plano elaborado de acordo com a demanda dos artefatos a serem produzidos.

A síndrome do *deadline* não é uma entidade ativa no ambiente ou no processo de desenvolvimento, *i.é*, ela geralmente acontece quando uma variável ou fator externo pertuba o ambiente, provocando erros de construções em pelo menos um artefato ou em tarefas específicas do processo de desenvolvimento de software através de construções *one-way*[2].

A sídrome do *deadline* é uma fenômeno que remete as pessoas e os times de desenvolvimento a um plano de trabalho que pode ser entendido como: “Não importa o processo, enquanto tiver tempo para a tarefa corrente, priorize as tarefas de menor *deadline*!”.

### 3.1. A Síndrome do *Deadline* X Síndrome do Estudante

A síndrome do estudante foi proposta originalmente por [Goldratt, 1997] através da aplicação da TOC no contexto do gerenciamento de projetos com o emprego da metodologia da corrente crítica ou método CCPM *(Critical Chain Project Management)*. Segundo [Leach, 2000], a síndrome do estudante é resultado do comportamento humano e sua principal característica é a procrastinação de uma atividade até um estado de urgência ou até o tempo limite para iniciar uma tarefa ou trabalho.

A figura 5 ilustra as características e os efeitos da síndrome do estudante, apresentando uma relação sobre o que foi planejado e o que verdadeiramente acontence na prática.

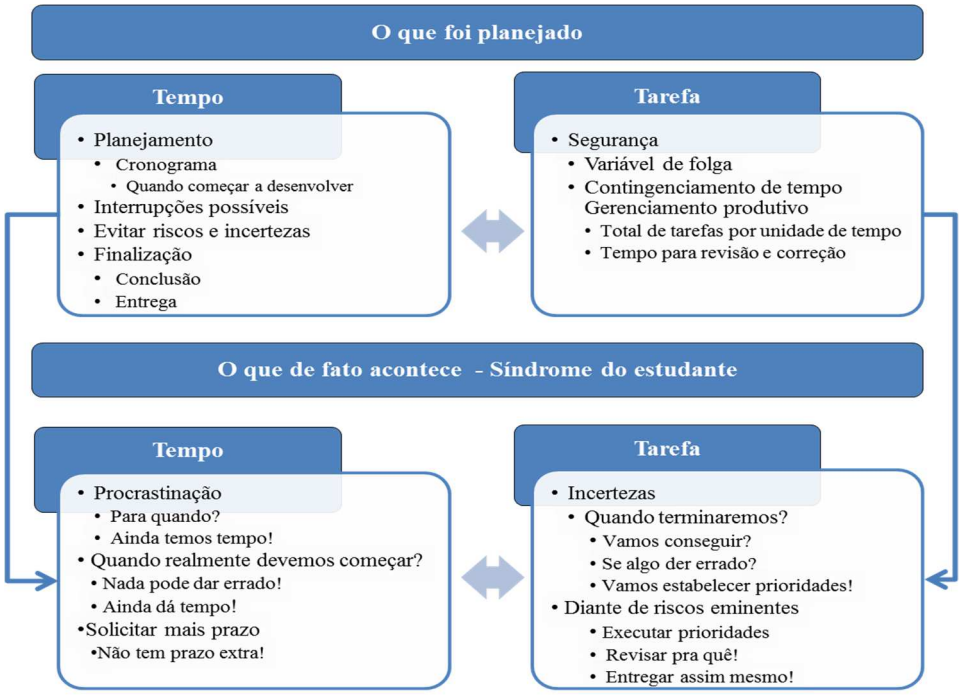


**Figura 5. Características e efeitos da síndrome do estudante**

[2] *One-way: Termo adotado para atividades desenvolvidas sem depuração (testes de V & V).*

Normalmente a síndrome do estudante não apresenta nenhum aspecto positivo em relação à execução do projeto a não ser pelo fato da capacidade de reação das pessoas diante de uma situação de urgência, onde o máximo de esforço é desprendido para realizar uma determinada tarefa. É o que representa a figura 6 a seguir, onde pode ser notado que os aspectos negativos sobrepõem o único fator positivo em uma relação (tempo x esforço).

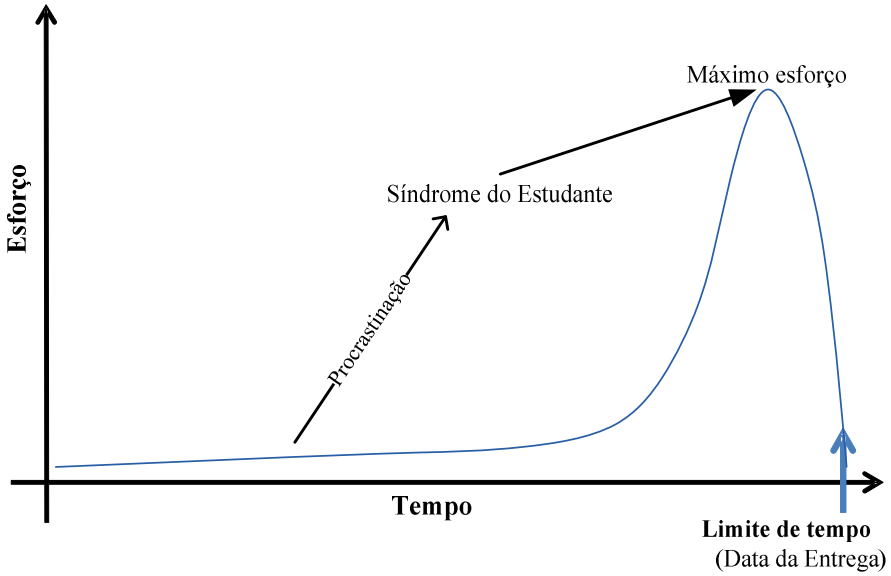


**Figura 6. Características e efeito da síndrome do estudante (adaptado de [Leach 2000])**

A síndrome do *deadline* difere da síndrome do estudante pelo fato de que a postergação ou o adiamento da execução de uma tarefa não se dá somente pela procrastinação e sim e principalmente pelas variáveis externas que perturbam o ambiente. Por este motivo, foi definido e adotado este termo diante da observação frente ao comportamento das equipes no desenvolvimento das tarefas próximo aos prazos de entrega.

As principais variáveis identificadas que perturbam o ambiente estão diretamente relacionadas com as atividades acadêmicas, como: provas, trabalhos, disponibilidade de recursos (laboratórios) e eventos acadêmicos, o que inclui: feiras, seminários, simpósios, palestras, etc. Ou até mesmo tarefas que fazem parte do escopo do projeto, mas que por razões de gerenciamento não foram escalonadas e não devem ser priorizadas, seja por ordem de precedência ou por restrições de recursos.

Um exemplo ilustrativo deste caso é a execução paralela de tarefas que não cobrem o estágio corrente do projeto: Por exemplo: codificar telas antes de codificar a arquitetura e camada de aplicação. Este tipo de situação, além de impactar no estágio atual do projeto, também pode implicar em problemas futuros de construto e retrabalho.

É importante ressaltar que o principal aspecto da síndrome do estudante, também está inserido no contexto da síndrome do *deadline*. A diferença básica é que os atrasos não são somente em virtude da procrastinação. Além disso, a observação a cerca da síndrome do *deadline* evidenciou que os times de desenvolvimento diante do prazo final de entrega, voltavam seus esforços para um único motivo ou razão: “entregar qualquer coisa no prazo!”. Isto acarreta impactos negativos no produto, principalmente em relação à qualidade, e no processo com retrabalhos e reescalonamento de tarefas.

A figura 7 é uma representação da síndrome do *deadline, com* suas características e efeitos no PDS provocada pelas influências externas.

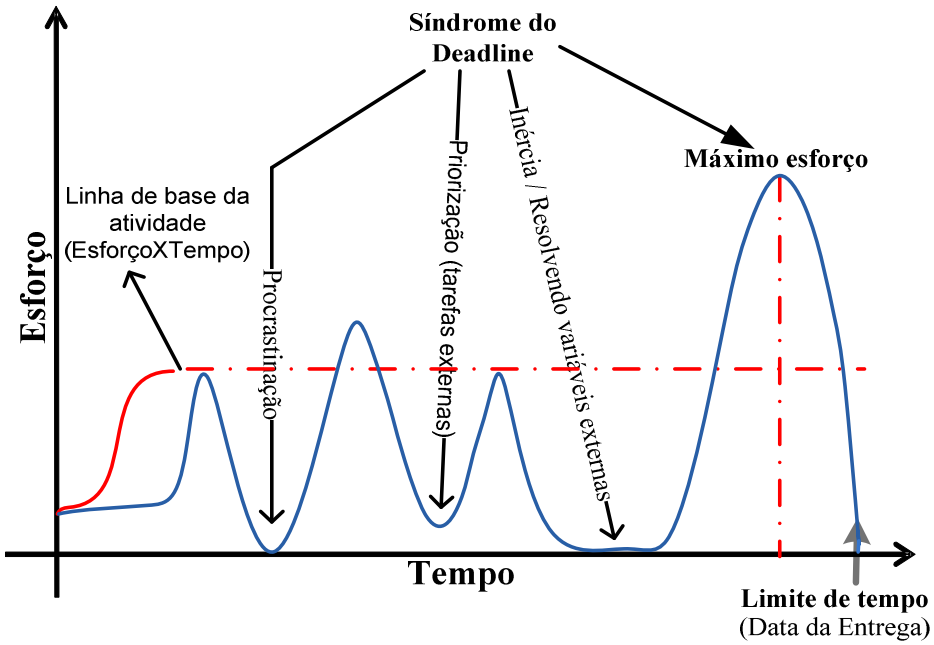


**Figura 7. Características e efeitos da síndrome do *deadline***

Acredita-se que na indústria de software também exista fatores externos relacionados ao cotidiano do gerenciamento de projetos que implicam diretamente nas equipes de desenvolvimento de software. Como consequência, estes fatores também impactam no processo e no desenvolvimento do produto final. Algumas destas implicações podem ser facilmente identificadas e listadas, como: (1) reuniões não planejadas ou não agendadas previamente; (2) necessidade de treinamentos não previstos; (3) falhas técnicas em hardware e software que afetam a produtividade das equipes; (4) mudanças não previstas no processo ou o produto, e (5) mudança ou troca de membros da equipe. Além é claro da priorização de tarefas sem prioridades ou não programadas, como aconteceu no experimento acadêmico.

### 3.2. A Síndrome do *Deadline* X Lei de Parkinson

Como visto, a sindrome do deadline não é resultado apenas da procrastinação até o limite máximo de tempo para poder realizar um trabalho. Da mesma forma que não é uma expansão do trabalho de modo a preencher todo o orçamento de tempo disponível para realizá-lo. Portanto, a síndrome do *deadline* também difere da principal característica da Lei de Parkinson, que é: "O trabalho se expande de modo a preencher todo o tempo disponível para a sua realização!" [Parkinson e Osborn, 1957].

Para melhor entendimento da observação de Parkinson, pode-se ater ao seguinte exemplo: se um membro da equipe tiver uma janela de 72 horas para executar uma tarefa que demora apenas 36 horas para ser realizada, ele irá intuitivamente tornar a tarefa complexa ao ponto de terminá-la ao fim das 72 horas, podendo ainda arrolar outros membros da equipe para ajudar na conclusão da mesma. Com isso, a produtividade é naturalmente afetada pela ineficácia da execução de uma tarefa. [Gutierrez e Kouvelis, 1991].

Segundo [Buchanan, 2009], a variante da Lei de Parkinson não é o tempo de folga e sim a eficácia com que a tarefa é executada em detrimento do tempo disponível. Desta forma, [Buchanan, 2009] infere que "quando se há menos tempo, a tendência é que sejamos mais eficazes, prático e objetivo."A figura 8 adaptada de [Parkinson e Osborn, 1957] e [Murray e Parkinson, 1958], apresenta uma correlação de esforço X tempo, dado pela execução de uma tarefa em uma agenda de tempo.

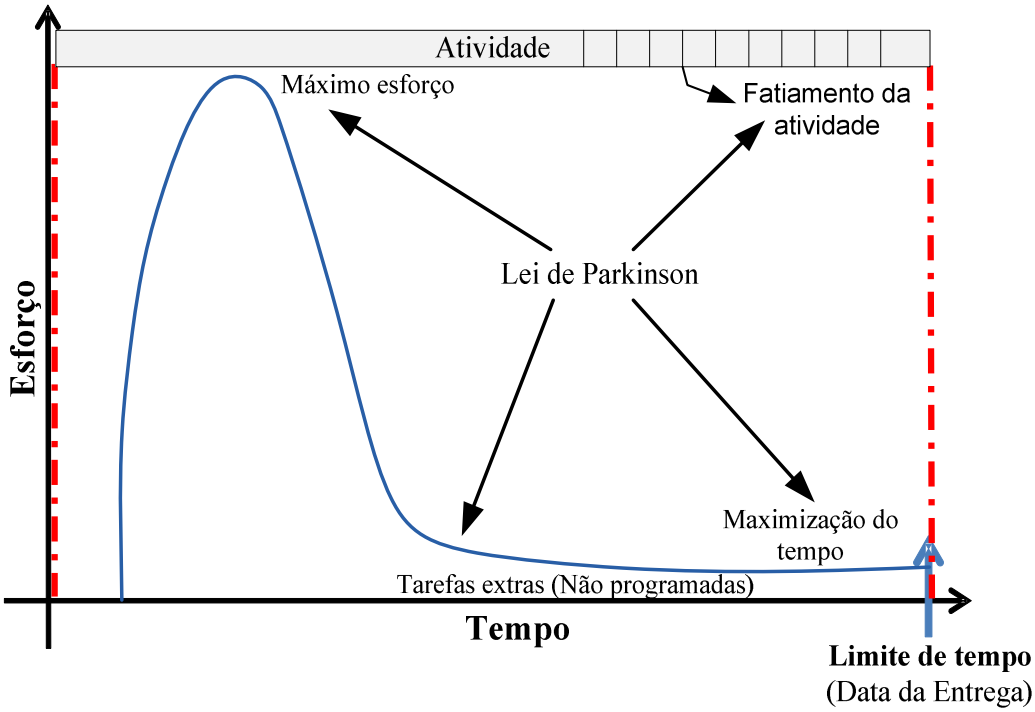


**Figura 8. Características e efeitos da Lei de Parkinson**

Na figura pode-se observar que a tarefa é "esticada" para cobrir toda a escala de tempo, mesmo que no inicio de realização da tarefa tenha sido consumido um maior esforço e tenha-se concluído a maior parte do trabalho.

## 4. Implicações da Síndrome do *Deadline* no PDS

O ambiente de desenvolvimento está diretamente associado aos problemas provocados pela síndrome e as perturbações ambientais interferem diretamente na produção da equipe de desenvolvimento. Ao contrário da síndrome do estudante, a síndrome do *deadline* apresenta outros fatores "positivos", além da percepção da capacidade de reação das pessoas envolvidas no processo.

A síndrome do *deadline* implica nos seguintes fatores positivos:

- Quando se conhece os fatores externos, permite-se o gerenciamento para a redução dos impactos;
- Melhora a redistribuição das tarefas pelas equipes de desenvolvimento;
- Apresenta melhoria na comunicação entre os membros da equipe;
- Melhora na relação interpessoal e na interação entre os indivíduos do grupo;
- Aumenta a busca pelo conhecimento e nas diversas formas de tratar o problema;
- Permite a identificação de caminhos otimizantes para resolver os problemas de construção dos artefatos.

Os pontos negativos, em geral seguem os mesmos observados pela síndrome do estudante, e as implicações mais comuns percebidas foram:

- Aumento do esforço durante a reta final do tempo de entrega;
- Construções incorretas ou imperfeitas;
- Falta de revisão das tarefas processadas / concluídas;
- Atividades não entregues ou entregas realizadas parcialmente.

Em uma execução industrial certamente poderia ser elencada ainda o ‘estouro no cronograma’ que tem como consequência o impacto financeiro no projeto, dentre outros fatores.

### 4.1 Gerenciamento e Controle da Síndrome do *Deadline*

Um contraponto da síndrome do deadline em relação à síndrome do estudante é a possibilidade de gerenciamento da síndrome e seus efeitos.

Nesta pesquisa a síndrome foi detectada no experimento 1 e teve impacto direto nas atividades, principalmente na segunda fase.

No segundo experimento foram promovidas algumas ações junto aos grupos na eminência de atenuar os problemas causados pela síndrome. Vale ressaltar que isso só foi possível devido ao controle exercido do interlocutor com a equipe, como monitoramento, acompanhamento *in loco e* supervisão.

No terceiro experimento o processo foi novamente revisado e ajustado, foi elaborada uma identificação e análise simples de riscos em função dos impactos ambientais provocados pelas variáveis externas. Todo o replanejamento foi possível diante de dois fatores: (1) identificação com antecedência os fatores externos, e (2) o histórico registrado no livro do experimento (*log book*). Diante disto, foi elaborado um cronograma que dirimia os conflitos com as perturbações externas.

A figura 9 a seguir apresenta um conjunto de medidas que podem ser tomadas para atenuar os danos causados pela síndrome.

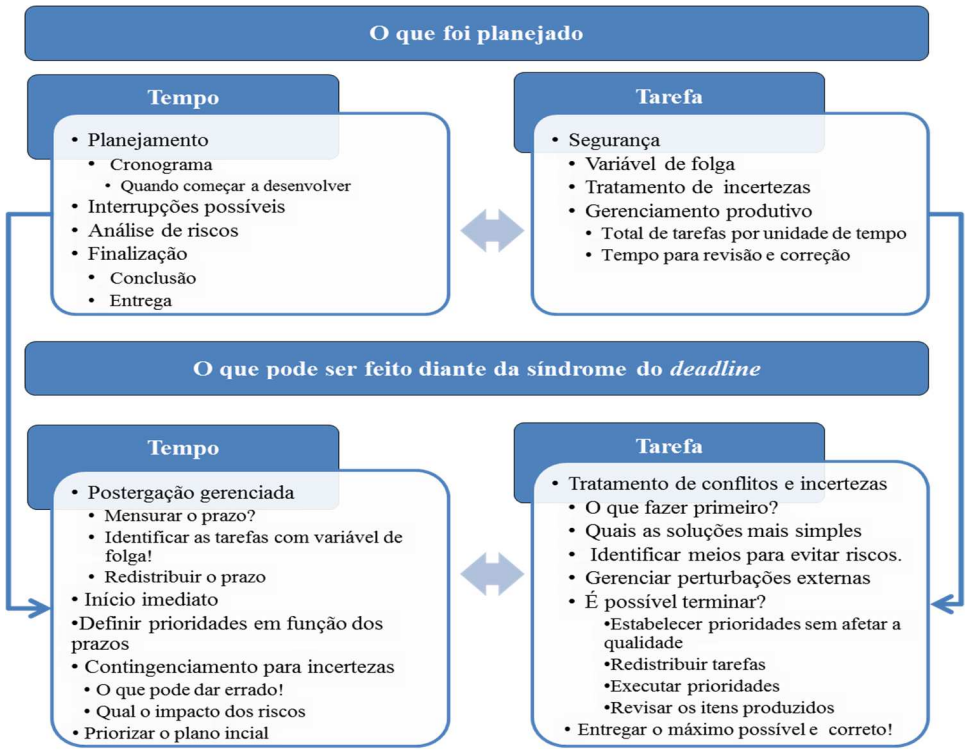


**Figura 9: Planejamento, gerenciamento e controle da síndrome do *deadline***

O problema é inevitável, mas pode ser atenuado com um bom planejamento e gerenciamento. Além disso, medidas como: (1) intensificação das cobranças; (2) orientação com relação à importância do projeto, (3) exigências para que as tarefas sejam entregues completas e dentro do prazo; e (4) rigidez no controle de qualidade, podem reduzir muito os impactos da síndrome no projeto e consequentemente no produto de software.

### 4.2 Causas da Síndrome do *Deadline*

A principal causa da síndrome do *deadline* está relacionada com o ambiente onde é executado o experimento.

Em virtude da execução deste projeto ser um ambiente acadêmico, os alunos foram sujeitos a interferências externas, como:

- Compromissos com outras disciplinas, trabalhos e provas;
- Semana de eventos e jornadas programada no calendário acadêmico;
- Interesse particular do time ou de parte do time no projeto;
- Acúmulo de tarefas causado por ausências e atrasos;
- Procrastinação das tarefas, conforme síndrome do estudante;
- Priorização de uma atividade não predecessora em detrimento a outra atividade que é predecessora;
- Desmotivação e inconsistências devido à falta de experiência em executar um projeto tal como na indústria.

### 4.3 Principais Problemas Causados pela Síndrome do *Deadline*

A observação e as anotações do livro de registro dos experimentos (*log book*), principalmente no experimento 1, evidenciaram potenciais problemas causados pela síndrome do *deadline*. Tais problemas acarretam em prejuízos produtivos para as equipes de desenvolvimento e, por conseguinte para o desenvolvimento do produto. Os principais problemas identificados são elencados, foram:

1. **Aumento dos conflitos interno:** os indivíduos entram em rota de colisão mediante as cobranças e a pressão para cumprir os prazos estabelecidos;
2. **Construção *One way*:** o termo "construção *one way*", foi criado para definir a construção sem verificação, sem validação ou atualizações, principalmente na documentação e nas plantas do sistema;
3. **Compensação:** as equipes compensam os atrasos com horas absurdas de trabalho contínuo, chegando alguns casos ter picos de 18 e até 20 horas seguidas.

### 4.4 Restrições no PDS Provocadas pela Síndrome do *Deadline*

As restrições encontradas oriundas da síndrome do deadline foram separadas em dois grupos: (1) restrições de ordem comportamental; e (2) restrições de ordem técnica.

As restrições apontadas na tabela 2 representam as restrições qualitativas de ordem comportamental encontradas nos *shops* dos Experimentos 1, 2 e 3.

**Tabela 2: Restrições de ordem comportamental provocadas pela *Síndrome do Deadline***

| **Restrições de Ordem Comportamental** | | | | |
|---|---|---|---|---|
| ***Id*** | ***Restrições*** | ***Exp 1*** | ***Exp 2*** | ***Exp 3*** |
| RC1 | Falha na Comunicação (Negligência e omissão da priorização) | Sim | Não | Sim |
| RC2 | Relacionamento interno (Aumento de Conflitos) | Sim | Sim | Não |
| RC3 | Desvio de foco: Impacto externo de outras atividades curriculares | Sim | Sim | Sim |
| RC4 | Cronograma negligenciado | Sim | Sim | Sim |

As restrições de ordem técnica estão diretamente relacionadas ao construto. Isto é, estão associadas ao descumprimento técnico das tarefas a serem realizadas.

A tabela 3 a seguir apresenta as restrições qualitativas de ordem técnica, encontradas nos *shops* dos Experimentos 1, 2 e 3.

**Tabela 3: Restrições de ordem técnicas provocadas pela Síndrome do Deadline**

| **Restrições de Ordem Técnica** | | | | |
|---|---|---|---|---|
| ***Id*** | ***Restrições*** | ***Exp 1*** | ***Exp 2*** | ***Exp 3*** |
| RT1 | Falta de revisão dos modelos | Sim | Sim | Sim |
| RT2 | Código não depurado | Sim | Sim | Sim |
| RT3 | Desobediência ao processo | Sim | Sim | Sim |
| RT4 | Desobediência aos padrões | Sim | Sim | Sim |
| RT5 | Testes mal especificados | Sim | Sim | Sim |
| RT6 | Testes não realizados | Sim | Sim | Sim |

## 4.5 Análise e Tratamento e das Restrições

As restrições encontradas foram analisadas, catalogadas e registradas no livro de registro do experimento (l*og book*). Isto foi feito de acordo com sua especificidade e em tempo de execução.

O tratamento foi sistematicamente aplicado em cada fase do PU ou tão logo que um problema foi detectado. Isto é importante, pois ajuda a dirimir os impactos no desenvolvimento do software.

A tabela 4 apresenta os grupos e as fases de ocorrências das restrições de ordem comportamental dos três experimentos.

**Tabela 4: Identificação, análise e tratamento das restrições de ordem comportamentais (RC).**

| **Avaliação das Restrições Comportamentais e Tratamento Aplicado** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Restrições** | **Grupos de Ocorrência** | | | **Fase de Ocorrência** | | | | **Tratamento** |
| | ***Exp 1*** | ***Exp 2*** | ***Exp 3*** | ***F1*** | ***F2*** | ***F3*** | ***F4*** | |
| ***RC1***[3] | G3 e G9 | G1 | - | - | - | X | - | Orientação e Cobrança |
| ***RC2*** | G9 | G1 | G2 | | X | X | X | Orientação, Motivação, Penalidades |
| ***RC3*** | G2, G4, G7 e G8 | G4, G8 e G9 | G1, G4 e G5 | - | X | - | - | Redistribuição igualitária das tarefas, Cobrança e Motivação. |
| ***RC4*** | G3 | G1 | G2 | | X | X | X | Orientação, Cobrança e Motivação |

O mesmo processo foi aplicado nas restrições de ordem técnicas. Além da cobrança, orientação, motivação, os grupos também receberam: treinamentos específicos sobre ferramentas e processos, aulas expositivas, *feedback* com os problemas encontrados e reuniões pontuais e individualizadas para tratar questões técnicas e de alinhamento com o processo. Este procedimento foi necessário para que as restrições técnicas não se transformassem em um gargalo produtivo.

A tabela 5 apresenta os grupos e as fases de ocorrências das restrições de ordem técnica dos experimentos 1, 2 e 3.

[3] *A RC2 foi evidenciada na terceira fase do Exp 1 e na segunda, terceira e quarta fase do Exp 2. A RC4 foi evidenciada nos três experimentos isto foi observado a partir da segunda fase.*

**Tabela 5: Identificação, análise e tratamento das restrições de ordem técnicas (RT).**

| Avaliação das Restrições Técnicas e Tratamento Aplicado | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Restrições** | **Grupos de Ocorrência** | | | **Fase de Ocorrência** | | | | **Tratamento** |
| | ***Exp 1*** | ***Exp 2*** | ***Exp 3*** | ***F1*** | ***F2*** | ***F3*** | ***F4*** | |
| ***RT1*** | G1, G3, G4 G7 e G9 | G4 | G2 | - | X | - | X | *Feedback* escrito, Treinamento/aula, Motivação e cobrança |
| ***RT2*** | G3, G1, G7 e G4 | G1 e G6 | G1, G4 | - | - | X | X | Orientação e Cobrança, *Feedback* escrito |
| ***RT3*** | G1, G3 | G1 e G6 | G5 | X | X | | | Orientação, Treinamento/aulas e Cobrança. |
| ***RT4*** | G3 | G1 e G6 | G2 | X | X | | | Cobrança, Treinamento/aulas, *Feedback* oral e escrito. |
| ***RT5*** | Todos (exceto G8) | Todos | G2 | - | - | X | X | Treinamento/aulas, Motivação e Cobrança, |
| ***RT6*** | G1, G3 e G4 | G1 e G6 | G2 e G5 | - | - | X | X | Feedback escrito e Cobrança |

## 5. Considerações Sobre a Síndrome do *Deadline*

A Síndrome do *Deadline*, suas causas e impactos foram observados nas três rodadas experimentais. No experimento 2 e no experimento 3, os impactos foram dirimidos, pois diante do histórico registrado e das lições aprendidas do experimento 1, o processo foi ajustado e o cronograma de entregas foi construído em função das perturbações externas identificadas no experimento 1. As restrições internas, onde houve priorizações de tarefas irrelevantes ou que não estavam programadas para serem executadas em um determinado estágio do processo, foram identificadas e tratadas aplicando a TOC.

Ainda assim, o experimento 2 e o experimento 3 apresentaram problemas causados pela síndrome. No experimento 3, foi detectado um problema de construção *one way* em um dos times de desenvolvimento. E quando estudado e averiguado o problema, constatou-se que a equipe havia antecipado todas as atividades de codificação da fase 3, conforme o *PU-Like*, em virtude de um evento em que iriam participar. Neste caso o grupo ainda tinha uma janela de duas semanas para concluir as tarefas da fase, mas fez todo o trabalho com a metade do tempo orçado, para que pudessem participar do evento. O resultado obviamente foi uma construção mal feita, incompleta e incorreta da tarefa e sem as devidas verificações e validações da produção. Isso chamou a atenção porque foi um fenômeno diferente do que estavam sendo rotineiramente observado. Pois a equipe comprimiu o tempo orçado antecipando e alocando todo trabalho um período muito curto. Normalmente o que acontece é a equipe usar o tempo disponível para priorizar o evento externo, deixando as tarefas do PDS para serem executas no limite da data da entrega.

A validação da síndrome do *deadline* ainda requer mais estudos e mais observações, principalmente em situações observadas na indústria de software. Mas como todo ambiente sofre de influências externas, é seguro dizer que este fenômeno também se aplica em outros ambientes, sobretudo nas fábricas de software. A diferença pode estar no tipo de impacto que a síndrome pode causar no processo e no produto, uma vez que ela é gerenciável, desde que conhecido os elementos causadores. Outro ponto, é que o gerenciamento nestes ambientes é feito com mais rigor porque geralmente a organizações dispõem de mais recursos para executar o projeto e possui um processo definido e maturado, além de equipes treinadas e pessoas responsáveis pelo gerenciamento do processo e dos riscos do projeto. Este trabalho mostrou que a síndrome do *deadline* é um problema da gestão de projetos e seus prejuízos dependem muito do planejamento, da execução e da equipe de desenvolvimento. Aparentemente, os problemas causados por este fenômeno tendem a ser diluídos com a maturidade do processo, do responsável pela condução do projeto e também pela repetição, que foi o caso deste estudo.

## 7. Referências


Carver, J.; Jaccheri, L.; Morasca, S.; (2003), Issues in Using Students in Empirical Studies in Software Engineering Education. Proceedings of the Ninth International Software Metrics Symposium (METRICS'03) 2003, IEEE Computer Society.

Buchanan, M. (2009), Parkinson's law revisited; Elsevier New Scientist Journal, Volume 201, Issue 2690, 7 January 2009, Pages 38–39, Published by Reed Business Information Ltd. London, England, UK.

Goldratt, E. M.; Cox, J.; (2003/06), The Goal: A Process of Ongoing Improvement; 2nd. Edition; Great Barrington, MA.; North River Press, 2003 Reprint 2006.

Goldratt, Eliyahu M.; (1997), Critical Chain. Great Barrington, MA: North River Press.

Goldhaber, A. S., Nieto, M. M.; (2010), "Photon and graviton mass limits", Rev. Mod. Phys. (American Physical Society), DOI:10.1103/RevModPhys.82.939; Pages 939–979.

Gupta, A., Bhardwaj, A., Kanda, A.; (2010), Fundamental Concepts of Theory of Constraints: An Emerging Philosophy; World Academy of Science, Engineering and Technology pp 686-692.

Gutierrez, G. J., Kouvelis**,** P.; (1991), Parkinson's Law and Its Implications for Project Management, Published by School of Business, Management Department, University of Texas at Austin, Austin, Texas USA.

Jacobson, I., Booch, G., Rumbaugh, J.; (1999), The Unified Software Development Process, Ed. Addison-Wesley Professional 1st Edition.

Kim, S.; Mabin, V. J.; Davies, J.; (2008), The theory of constraints thinking processes: retrospect and prospect; International Journal of Operations & Production Management; Vol. 28, Nr. 2 pp. 155-184, Emerald Group Publishing Limited; Wellington, New Zealand.

Larman, C.; (2004), Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development, Third Edition; Ed. Addison Wesley Professional.

Leach, L. P.; (2000), Critical Chain Project Management, First Edition, Ed. Artech House, Inc., Norwood - MA.

Murray, J., Parkinson, C. N.; (1958), Parkinson's Law or the Pursuit of Progress. Revival Books Ltd. Rossendale, LANCS, UK.

Pandit, S. V.; (2012), Application Of Theory Of Constraints On Scheduling Of Drum-Buffer-Rope System, IOSR Journal of Mechanical and Civil Engineering (IOSR-JMCE) Proceedings on Second International Conference on Emerging Trends in Engineering (SICETE).

Parkinson, C. N. (1955). Parkinson's Law, or The Pursuit of Progress, The Economist. Mont Clair - UK.

Parkinson, C, N., Osborn, R.C.; (1957), Parkinson's law (and other studies in administration), Seventeenth Printing, The Rise Press, Cambridge - Massachusetts, USA.

Ribeiro, S. A.; (2017); Identificação de Gargalos em Processo de Desenvolvimento de Software: Uma Proposta Baseada nos Princípios da Teoria das Restrições; Tese de Doutorado; Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais (UFRJ/PPGI).

Ribeiro, S. A.; Schmitz, E. A.; Alencar, A. J. S. M,; Silva, M. F.; (2017), Research Opportunities on the Application of the Theory of Constraints to Software Development Process; Journal of Software; IAP - International Academy Publishing. JSW-Vol 12. Nr. 4.

Sengupta, S.; Das, K., Vantil, R. P.; (2008), A New Method for Bottleneck Detection; Proceedings of the 2008 Winter Simulation Conference/IEEE.

Schragenheim, E., Dettmer H. W.; (2000), Drum-Buffer-Rope: A Whole System Approach to High Velocity Manufacturing: Optimizing Supply Chain Business Performance, St. Lucie Press Boca Raton, FL. USA.