

COLEÇÃO

# DESAFIOS DAS ENGENHARIAS:

ENGENHARIA DE COMPUTAÇÃO



ERNANE ROSA MARTINS  
(ORGANIZADOR)

Atena  
Editora  
Ano 2021

COLEÇÃO  
**DESAFIOS**  
DAS  
**ENGENHARIAS:**

**ENGENHARIA DE COMPUTAÇÃO**



**ERNANE ROSA MARTINS**  
(ORGANIZADOR)

**Atena**  
Editora  
Ano 2021

**Editora chefe**

Profª Drª Antonella Carvalho de Oliveira

**Assistentes editoriais**

Natalia Oliveira

Flávia Roberta Barão

**Bibliotecária**

Janaina Ramos

**Projeto gráfico**

Natália Sandrini de Azevedo

Camila Alves de Cremona

Luiza Alves Batista

Maria Alice Pinheiro

**Imagens da capa**

iStock

**Edição de arte**

Luiza Alves Batista

**Revisão**

Os autores

2021 by Atena Editora

Copyright © Atena Editora

Copyright do Texto © 2021 Os autores

Copyright da Edição © 2021 Atena Editora

Direitos para esta edição cedidos à Atena Editora pelos autores.

Open access publication by Atena Editora



Todo o conteúdo deste livro está licenciado sob uma Licença de Atribuição *Creative Commons*. Atribuição-Não-Comercial-NãoDerivativos 4.0 Internacional (CC BY-NC-ND 4.0).

O conteúdo dos artigos e seus dados em sua forma, correção e confiabilidade são de responsabilidade exclusiva dos autores, inclusive não representam necessariamente a posição oficial da Atena Editora. Permitido o *download* da obra e o compartilhamento desde que sejam atribuídos créditos aos autores, mas sem a possibilidade de alterá-la de nenhuma forma ou utilizá-la para fins comerciais.

Todos os manuscritos foram previamente submetidos à avaliação cega pelos pares, membros do Conselho Editorial desta Editora, tendo sido aprovados para a publicação com base em critérios de neutralidade e imparcialidade acadêmica.

A Atena Editora é comprometida em garantir a integridade editorial em todas as etapas do processo de publicação, evitando plágio, dados ou resultados fraudulentos e impedindo que interesses financeiros comprometam os padrões éticos da publicação. Situações suspeitas de má conduta científica serão investigadas sob o mais alto padrão de rigor acadêmico e ético.

**Conselho Editorial**

**Ciências Humanas e Sociais Aplicadas**

Prof. Dr. Alexandre Jose Schumacher – Instituto Federal de Educação, Ciência e Tecnologia do Paraná

Prof. Dr. Américo Junior Nunes da Silva – Universidade do Estado da Bahia

Profª Drª Andréa Cristina Marques de Araújo – Universidade Fernando Pessoa

Prof. Dr. Antonio Carlos Frasson – Universidade Tecnológica Federal do Paraná

Prof. Dr. Antonio Gasparetto Júnior – Instituto Federal do Sudeste de Minas Gerais

Prof. Dr. Antonio Isidro-Filho – Universidade de Brasília

Prof. Dr. Arnaldo Oliveira Souza Júnior – Universidade Federal do Piauí  
Prof. Dr. Carlos Antonio de Souza Moraes – Universidade Federal Fluminense  
Prof. Dr. Crisóstomo Lima do Nascimento – Universidade Federal Fluminense  
Profª Drª Cristina Gaio – Universidade de Lisboa  
Prof. Dr. Daniel Richard Sant’Ana – Universidade de Brasília  
Prof. Dr. Deyvison de Lima Oliveira – Universidade Federal de Rondônia  
Profª Drª Dilma Antunes Silva – Universidade Federal de São Paulo  
Prof. Dr. Edvaldo Antunes de Farias – Universidade Estácio de Sá  
Prof. Dr. Elson Ferreira Costa – Universidade do Estado do Pará  
Prof. Dr. Eloi Martins Senhora – Universidade Federal de Roraima  
Prof. Dr. Gustavo Henrique Cepolini Ferreira – Universidade Estadual de Montes Claros  
Prof. Dr. Humberto Costa – Universidade Federal do Paraná  
Profª Drª Ivone Goulart Lopes – Istituto Internazionele delle Figlie de Maria Ausiliatrice  
Prof. Dr. Jadson Correia de Oliveira – Universidade Católica do Salvador  
Prof. Dr. José Luis Montesillo-Cedillo – Universidad Autónoma del Estado de México  
Prof. Dr. Julio Candido de Meirelles Junior – Universidade Federal Fluminense  
Profª Drª Lina Maria Gonçalves – Universidade Federal do Tocantins  
Prof. Dr. Luis Ricardo Fernandes da Costa – Universidade Estadual de Montes Claros  
Profª Drª Natiéli Piovesan – Instituto Federal do Rio Grande do Norte  
Prof. Dr. Marcelo Pereira da Silva – Pontifícia Universidade Católica de Campinas  
Profª Drª Maria Luzia da Silva Santana – Universidade Federal de Mato Grosso do Sul  
Prof. Dr. Miguel Rodrigues Netto – Universidade do Estado de Mato Grosso  
Prof. Dr. Pablo Ricardo de Lima Falcão – Universidade de Pernambuco  
Profª Drª Paola Andressa Scortegagna – Universidade Estadual de Ponta Grossa  
Profª Drª Rita de Cássia da Silva Oliveira – Universidade Estadual de Ponta Grossa  
Prof. Dr. Rui Maia Diamantino – Universidade Salvador  
Prof. Dr. Saulo Cerqueira de Aguiar Soares – Universidade Federal do Piauí  
Prof. Dr. Urandi João Rodrigues Junior – Universidade Federal do Oeste do Pará  
Profª Drª Vanessa Bordin Viera – Universidade Federal de Campina Grande  
Profª Drª Vanessa Ribeiro Simon Cavalcanti – Universidade Católica do Rio de Janeiro  
Prof. Dr. William Cleber Domingues Silva – Universidade Federal Rural do Rio de Janeiro  
Prof. Dr. Willian Douglas Guilherme – Universidade Federal do Tocantins

#### **Ciências Agrárias e Multidisciplinar**

Prof. Dr. Alexandre Igor Azevedo Pereira – Instituto Federal Goiano  
Prof. Dr. Arinaldo Pereira da Silva – Universidade Federal do Sul e Sudeste do Pará  
Prof. Dr. Antonio Pasqualetto – Pontifícia Universidade Católica de Goiás  
Profª Drª Carla Cristina Bauermann Brasil – Universidade Federal de Santa Maria  
Prof. Dr. Cleberton Correia Santos – Universidade Federal da Grande Dourados  
Profª Drª Diocléa Almeida Seabra Silva – Universidade Federal Rural da Amazônia  
Prof. Dr. Écio Souza Diniz – Universidade Federal de Viçosa  
Prof. Dr. Fábio Steiner – Universidade Estadual de Mato Grosso do Sul  
Prof. Dr. Fágner Cavalcante Patrocínio dos Santos – Universidade Federal do Ceará  
Profª Drª Girlene Santos de Souza – Universidade Federal do Recôncavo da Bahia  
Prof. Dr. Jael Soares Batista – Universidade Federal Rural do Semi-Árido  
Prof. Dr. Jayme Augusto Peres – Universidade Estadual do Centro-Oeste  
Prof. Dr. Júlio César Ribeiro – Universidade Federal Rural do Rio de Janeiro  
Profª Drª Lina Raquel Santos Araújo – Universidade Estadual do Ceará  
Prof. Dr. Pedro Manuel Villa – Universidade Federal de Viçosa  
Profª Drª Raissa Rachel Salustriano da Silva Matos – Universidade Federal do Maranhão  
Prof. Dr. Ronilson Freitas de Souza – Universidade do Estado do Pará  
Profª Drª Talita de Santos Matos – Universidade Federal Rural do Rio de Janeiro

Prof. Dr. Tiago da Silva Teófilo – Universidade Federal Rural do Semi-Árido  
Prof. Dr. Valdemar Antonio Paffaro Junior – Universidade Federal de Alfenas

### **Ciências Biológicas e da Saúde**

Prof. Dr. André Ribeiro da Silva – Universidade de Brasília  
Profª Drª Anelise Levay Murari – Universidade Federal de Pelotas  
Prof. Dr. Benedito Rodrigues da Silva Neto – Universidade Federal de Goiás  
Profª Drª Daniela Reis Joaquim de Freitas – Universidade Federal do Piauí  
Profª Drª Débora Luana Ribeiro Pessoa – Universidade Federal do Maranhão  
Prof. Dr. Douglas Siqueira de Almeida Chaves – Universidade Federal Rural do Rio de Janeiro  
Prof. Dr. Edson da Silva – Universidade Federal dos Vales do Jequitinhonha e Mucuri  
Profª Drª Elizabeth Cordeiro Fernandes – Faculdade Integrada Medicina  
Profª Drª Eleuza Rodrigues Machado – Faculdade Anhanguera de Brasília  
Profª Drª Elane Schwinden Prudêncio – Universidade Federal de Santa Catarina  
Profª Drª Eysler Gonçalves Maia Brasil – Universidade da Integração Internacional da Lusofonia Afro-Brasileira  
Prof. Dr. Ferlando Lima Santos – Universidade Federal do Recôncavo da Bahia  
Profª Drª Fernanda Miguel de Andrade – Universidade Federal de Pernambuco  
Prof. Dr. Fernando Mendes – Instituto Politécnico de Coimbra – Escola Superior de Saúde de Coimbra  
Profª Drª Gabriela Vieira do Amaral – Universidade de Vassouras  
Prof. Dr. Gianfábio Pimentel Franco – Universidade Federal de Santa Maria  
Prof. Dr. Helio Franklin Rodrigues de Almeida – Universidade Federal de Rondônia  
Profª Drª Iara Lúcia Tescarollo – Universidade São Francisco  
Prof. Dr. Igor Luiz Vieira de Lima Santos – Universidade Federal de Campina Grande  
Prof. Dr. Jefferson Thiago Souza – Universidade Estadual do Ceará  
Prof. Dr. Jesus Rodrigues Lemos – Universidade Federal do Piauí  
Prof. Dr. Jônatas de França Barros – Universidade Federal do Rio Grande do Norte  
Prof. Dr. José Max Barbosa de Oliveira Junior – Universidade Federal do Oeste do Pará  
Prof. Dr. Luís Paulo Souza e Souza – Universidade Federal do Amazonas  
Profª Drª Magnólia de Araújo Campos – Universidade Federal de Campina Grande  
Prof. Dr. Marcus Fernando da Silva Praxedes – Universidade Federal do Recôncavo da Bahia  
Profª Drª Maria Tatiane Gonçalves Sá – Universidade do Estado do Pará  
Profª Drª Mylena Andréa Oliveira Torres – Universidade Ceuma  
Profª Drª Natiéli Piovesan – Instituto Federac do Rio Grande do Norte  
Prof. Dr. Paulo Inada – Universidade Estadual de Maringá  
Prof. Dr. Rafael Henrique Silva – Hospital Universitário da Universidade Federal da Grande Dourados  
Profª Drª Regiane Luz Carvalho – Centro Universitário das Faculdades Associadas de Ensino  
Profª Drª Renata Mendes de Freitas – Universidade Federal de Juiz de Fora  
Profª Drª Vanessa da Fontoura Custódio Monteiro – Universidade do Vale do Sapucaí  
Profª Drª Vanessa Lima Gonçalves – Universidade Estadual de Ponta Grossa  
Profª Drª Vanessa Bordin Viera – Universidade Federal de Campina Grande  
Profª Drª Welma Emidio da Silva – Universidade Federal Rural de Pernambuco

### **Ciências Exatas e da Terra e Engenharias**

Prof. Dr. Adélio Alcino Sampaio Castro Machado – Universidade do Porto  
Profª Drª Ana Grasielle Dionísio Corrêa – Universidade Presbiteriana Mackenzie  
Prof. Dr. Carlos Eduardo Sanches de Andrade – Universidade Federal de Goiás  
Profª Drª Carmen Lúcia Voigt – Universidade Norte do Paraná  
Prof. Dr. Cleiseano Emanuel da Silva Paniagua – Instituto Federal de Educação, Ciência e Tecnologia de Goiás  
Prof. Dr. Douglas Gonçalves da Silva – Universidade Estadual do Sudoeste da Bahia  
Prof. Dr. Eloi Rufato Junior – Universidade Tecnológica Federal do Paraná  
Profª Drª Érica de Melo Azevedo – Instituto Federal do Rio de Janeiro

Prof. Dr. Fabrício Menezes Ramos – Instituto Federal do Pará  
Profª Dra. Jéssica Verger Nardeli – Universidade Estadual Paulista Júlio de Mesquita Filho  
Prof. Dr. Juliano Carlo Rufino de Freitas – Universidade Federal de Campina Grande  
Profª Drª Luciana do Nascimento Mendes – Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte  
Prof. Dr. Marcelo Marques – Universidade Estadual de Maringá  
Prof. Dr. Marco Aurélio Kistemann Junior – Universidade Federal de Juiz de Fora  
Profª Drª Neiva Maria de Almeida – Universidade Federal da Paraíba  
Profª Drª Natiéli Piovesan – Instituto Federal do Rio Grande do Norte  
Profª Drª Priscila Tessmer Scaglioni – Universidade Federal de Pelotas  
Prof. Dr. Sidney Gonçalves de Lima – Universidade Federal do Piauí  
Prof. Dr. Takeshy Tachizawa – Faculdade de Campo Limpo Paulista

#### **Linguística, Letras e Artes**

Profª Drª Adriana Demite Stephani – Universidade Federal do Tocantins  
Profª Drª Angeli Rose do Nascimento – Universidade Federal do Estado do Rio de Janeiro  
Profª Drª Carolina Fernandes da Silva Mandaji – Universidade Tecnológica Federal do Paraná  
Profª Drª Denise Rocha – Universidade Federal do Ceará  
Profª Drª Edna Alencar da Silva Rivera – Instituto Federal de São Paulo  
Profª Drª Fernanda Tonelli – Instituto Federal de São Paulo,  
Prof. Dr. Fabiano Tadeu Grazioli – Universidade Regional Integrada do Alto Uruguai e das Missões  
Prof. Dr. Gilmei Fleck – Universidade Estadual do Oeste do Paraná  
Profª Drª Keyla Christina Almeida Portela – Instituto Federal de Educação, Ciência e Tecnologia do Paraná  
Profª Drª Miranilde Oliveira Neves – Instituto de Educação, Ciência e Tecnologia do Pará  
Profª Drª Sandra Regina Gardacho Pietrobon – Universidade Estadual do Centro-Oeste  
Profª Drª Sheila Marta Carregosa Rocha – Universidade do Estado da Bahia

## Coleção desafios das engenharias: engenharia de computação

**Diagramação:** Camila Alves de Cremo  
**Correção:** Maiara Ferreira  
**Indexação:** Gabriel Motomu Teshima  
**Revisão:** Os autores  
**Organizador:** Ernane Rosa Martins

### Dados Internacionais de Catalogação na Publicação (CIP)

C691 Coleção desafios das engenharias: engenharia de computação / Organizador Ernane Rosa Martins. - Ponta Grossa - PR: Atena, 2021.

Formato: PDF

Requisitos de sistema: Adobe Acrobat Reader

Modo de acesso: World Wide Web

Inclui bibliografia

ISBN 978-65-5983-387-0

DOI: <https://doi.org/10.22533/at.ed.870211808>

1. Engenharia da computação. I. Martins, Ernane Rosa (Organizador). II. Título.

CDD 621.39

Elaborado por Bibliotecária Janaina Ramos - CRB-8/9166

**Atena Editora**

Ponta Grossa - Paraná - Brasil  
Telefone: +55 (42) 3323-5493

[www.atenaeditora.com.br](http://www.atenaeditora.com.br)

[contato@atenaeditora.com.br](mailto:contato@atenaeditora.com.br)

## DECLARAÇÃO DOS AUTORES

Os autores desta obra: 1. Atestam não possuir qualquer interesse comercial que constitua um conflito de interesses em relação ao artigo científico publicado; 2. Declaram que participaram ativamente da construção dos respectivos manuscritos, preferencialmente na: a) Concepção do estudo, e/ou aquisição de dados, e/ou análise e interpretação de dados; b) Elaboração do artigo ou revisão com vistas a tornar o material intelectualmente relevante; c) Aprovação final do manuscrito para submissão.; 3. Certificam que os artigos científicos publicados estão completamente isentos de dados e/ou resultados fraudulentos; 4. Confirmam a citação e a referência correta de todos os dados e de interpretações de dados de outras pesquisas; 5. Reconhecem terem informado todas as fontes de financiamento recebidas para a consecução da pesquisa; 6. Autorizam a edição da obra, que incluem os registros de ficha catalográfica, ISBN, DOI e demais indexadores, projeto visual e criação de capa, diagramação de miolo, assim como lançamento e divulgação da mesma conforme critérios da Atena Editora.



## DECLARAÇÃO DA EDITORA

A Atena Editora declara, para os devidos fins de direito, que: 1. A presente publicação constitui apenas transferência temporária dos direitos autorais, direito sobre a publicação, inclusive não constitui responsabilidade solidária na criação dos manuscritos publicados, nos termos previstos na Lei sobre direitos autorais (Lei 9610/98), no art. 184 do Código penal e no art. 927 do Código Civil; 2. Autoriza e incentiva os autores a assinarem contratos com repositórios institucionais, com fins exclusivos de divulgação da obra, desde que com o devido reconhecimento de autoria e edição e sem qualquer finalidade comercial; 3. Todos os e-book são *open access*, desta forma não os comercializa em seu site, sites parceiros, plataformas de *e-commerce*, ou qualquer outro meio virtual ou físico, portanto, está isenta de repasses de direitos autorais aos autores; 4. Todos os membros do conselho editorial são doutores e vinculados a instituições de ensino superior públicas, conforme recomendação da CAPES para obtenção do Qualis livro; 5. Não cede, comercializa ou autoriza a utilização dos nomes e e-mails dos autores, bem como nenhum outro dado dos mesmos, para qualquer finalidade que não o escopo da divulgação desta obra.

## APRESENTAÇÃO

A Engenharia de Computação tem como definição ser o ramo da engenharia que se caracteriza pelo projeto, desenvolvimento e implementação de sistemas, equipamentos e dispositivos computacionais, segundo uma visão integrada de hardware e software, apoiando-se em uma sólida base matemática e conhecimentos de fenômenos físicos. O objetivo é a aplicação das tecnologias de computação na solução de problemas de Engenharia.

Deste modo, este livro, aborda diversos aspectos tecnológicos computacionais, tais como: o desenvolvimento de um jogo de RPG acessível em LIBRAS; uma reflexão quanto à necessidade de aplicação de supressores de surto como proteção de transformadores devido a eventos transitórios em manobras de disjuntores; um algoritmo para geração de contorno 2D envolvendo regiões irregulares; avaliação da influência das tensões residuais e imperfeições geométricas iniciais em colunas de aço submetidas à flexão em torno do eixo de menor inércia; os esforços em estruturas laminares, de características de geometria e carregamentos diversos através da implementação computacional de um elemento finito sólido hexaédrico de 8 nós programado com uma linguagem computacional de alto nível; uma análise computacional realizada através do programa SAP2000; a estabilidade e as vibrações de anéis e tubulações apoiados em uma fundação elástica de Pasternak; um controlador neural para dois elos de um robô manipulador de três graus de liberdade (3 GDL); uma ferramenta de autoria para livros relacionados a área da educação; um aplicativo com propósito de aumentar a taxa de reciclagem e minimizar os danos ambientais devido ao descarte incorreto de resíduos na natureza; a conscientização de crianças e adolescentes sobre as ocorrências de bullying; uma aplicação web interativa, de fácil utilização e interface amigável, por meio do pacote Shiny, destinada aos tópicos de intervalo de confiança e dimensionamento de amostra para o parâmetro proporção; segmentar e detectar, por meio de redes neurais convolutivas, as pás dos raspadores de escória em painéis de ferro gusa do Reator Kambara de uma siderúrgica; integrar a Biblioteca Digital de Artigos (IFPublica) e a Plataforma de Digital de Inscrição e Administração de Projetos (PDIAP), por meio de adaptações nos dois projetos, para impedir erros humanos e automatizar o processo de cadastro de artigos do PDIAP na base de dados do IFPublica.

Assim, espero que a presente obra venha a se tornar um guia aos estudantes e profissionais da área de Engenharia de Computação, auxiliando-os em diversos assuntos relevantes da área, fornecendo a estes novos conhecimentos para poderem atender as necessidades informacionais, computacionais e de automação das organizações de uma forma geral. Por fim, agradeço aos autores por suas contribuições na construção desta importante obra e desejo muito sucesso a todos os nossos leitores.

## SUMÁRIO

### **CAPÍTULO 1..... 1**

A ELASTO-PLASTIC CONSTITUTIVE MODEL BASED ON CHABOCHE KINEMATIC HARDENING OF ALUMINUM ALLOY 7050-T7451

Renzo Fernandes Bastos

Daniel Masarin

Ernesto Massaroppi Junior

 <https://doi.org/10.22533/at.ed.8702118081>

### **CAPÍTULO 2..... 11**


AGANNO: UM JOGO DE RPG COM UMA PROPOSTA DE ACESSIBILIDADE USANDO LIBRAS

Gabriel Barroso da Silva Lima

Marcos Roberto dos Santos

Almir de Oliveira Costa Junior

Jucimar Maia da Silva Junior

 <https://doi.org/10.22533/at.ed.8702118082>

### **CAPÍTULO 3..... 23**

A IMPORTÂNCIA ATUAL DE ESTUDOS DE TRANSITÓRIOS ELETROMAGNÉTICOS PARA DEFINIÇÃO DE SISTEMAS DE PROTEÇÃO DE TRANSFORMADORES CONTRA SOBRETENSÕES E AS APLICAÇÕES RECENTES COM A INSTALAÇÃO DE SUPRESSORES DE SURTO

Nelson Clodoaldo de Jesus

João Roberto Cogo

Luiz Marlus Duarte

Luis Fernando Ribeiro Ferreira

Éverson Júnior de Mendonça

Leandro Martins Fernandes

 <https://doi.org/10.22533/at.ed.8702118083>

### **CAPÍTULO 4..... 38**

ALGORITMO PARA GERAÇÃO DE CONTORNO DE MALHAS RETANGULARES PARA CÁLCULO DE DIFERENÇAS FINITAS

Pedro Zaffalon da Silva


Neyva Maria Lopes Romeiro

Rafael Furlanetto Casamaximo

Iury Pereira de Souza

Paulo Laerte Natti

Eliandro Rodrigues Cirilo


 <https://doi.org/10.22533/at.ed.8702118084>

### **CAPÍTULO 5..... 53**

ANÁLISE DA RESISTÊNCIA DE PILARES DE AÇO SOB A INFLUÊNCIA DE TENSÕES RESIDUAIS E IMPERFEIÇÕES GEOMÉTRICAS INICIAIS

Jefferson Alves Ferreira


Giovani Vitório Costa  
Harley Francisco Viana  
Renata Gomes Lanna da Silva

 <https://doi.org/10.22533/at.ed.8702118085>

**CAPÍTULO 6..... 70**

**ANÁLISE DE ESTRUTURAS LAMINARES UTILIZANDO UM ELEMENTO SÓLIDO DE BAIXA ORDEM ENRIQUECIDO COM MODOS INCOMPATÍVEIS**


Erijohnson da Silva Ferreira  
William Taylor Matias Silva  
Sebastião Simão da Silva  
Adenilda Timóteo Salviano  
José Lucas Pessoa de Oliveira

 <https://doi.org/10.22533/at.ed.8702118086>

**CAPÍTULO 7..... 84**

**ANÁLISE ESTRUTURAL DO EDIFÍCIO SEDE DA PROCURADORIA GERAL DA REPÚBLICA: O ESTUDO DE CASO DO BLOCO “A”**


Stefano Galimi  
Márcio Augusto Roma Buzar  
Marco Aurélio Bessa  
Leonardo da Silveira Pirillo Inojosa

 <https://doi.org/10.22533/at.ed.8702118087>

**CAPÍTULO 8..... 103**

**ANÁLISE ESTRUTURAL DO EDIFÍCIO SEDE DA PROCURADORIA GERAL DA REPÚBLICA: O ESTUDO DE CASO DO BLOCO “B”**


Stefano Galimi  
Márcio Augusto Roma Buzar  
Marco Aurélio Bessa  
Marcos Henrique Ritter de Gregorio

 <https://doi.org/10.22533/at.ed.8702118088>

**CAPÍTULO 9..... 119**

**APPLICATION OF A MULTI-OBJECTIVE OPTIMIZATION PARETO APPROACH TO DESIGN THE SDRE CONTROLLER FOR A RIGID-FLEXIBLE SATELLITE**

Luiz Carlos Gadelha de Souza







 <https://doi.org/10.22533/at.ed.8702118089>







**CAPÍTULO 10..... 131**

**APPLICATION OF DEEP LEARNING FOR ANALYSIS OF CRACKS IN PELLET FALLING TESTS**

Marconi Junio Henriques Magnani  
Jorge José Fernandes Filho  
Thyago Rosa Souza  
Marco Antonio de Souza Leite Cuadros

 <https://doi.org/10.22533/at.ed.87021180810>

<b>CAPÍTULO 11</b> .....	<b>143</b>
FLAMBAGEM E VIBRAÇÃO DE ANÉIS E TUBULAÇÕES ESBELTAS EM UMA FUNDAÇÃO ELÁSTICA	
Mariana Barros dos Santos Dias Paulo Batista Gonçalves	
 <a href="https://doi.org/10.22533/at.ed.87021180811">https://doi.org/10.22533/at.ed.87021180811</a>	
<b>CAPÍTULO 12</b> .....	<b>155</b>
CALIDAD ÁGIL: PATRONES DE DISEÑO EN UN CONTEXTO DE DESARROLLO DIRIGIDO POR PRUEBAS	
Anna Grimán Padua Manuel Capel Tuñón Eladio Garví	
 <a href="https://doi.org/10.22533/at.ed.87021180812">https://doi.org/10.22533/at.ed.87021180812</a>	
<b>CAPÍTULO 13</b> .....	<b>168</b>
CONTROLE NEURAL DE DOIS ELOS DE UM ROBÔ DE TRÊS GRAUS DE LIBERDADE	
José Antonio Riul Paulo Henrique de Miranda Montenegro	
 <a href="https://doi.org/10.22533/at.ed.87021180813">https://doi.org/10.22533/at.ed.87021180813</a>	
<b>CAPÍTULO 14</b> .....	<b>181</b>
SUBOPTIMAL CONTROL ON NONLINEAR SATELLITE SIMULATIONS USING SDRE AND H-INFINITY	
Alessandro Gerlinger Romero Luiz Carlos Gadelha de Souza	
 <a href="https://doi.org/10.22533/at.ed.87021180814">https://doi.org/10.22533/at.ed.87021180814</a>	
<b>CAPÍTULO 15</b> .....	<b>193</b>
CREATE REALITY IN BOOKS (CRINB) - PROPOSTA DE FERRAMENTA DE AUTORIA DE LIVROS COM REALIZADADE AUMENTADA	
Lucas Velho Gomes Felipe Zunino Gabriel Abreu Freire Sidney Ferreira Coutinho Rogério Grijo Biazotto Eduardo Henrique Gomes Nelson Nascimento Júnior	
 <a href="https://doi.org/10.22533/at.ed.87021180815">https://doi.org/10.22533/at.ed.87021180815</a>	
<b>CAPÍTULO 16</b> .....	<b>198</b>
DESENVOLVIMENTO DE ATIVIDADES DE ORIENTAÇÃO E CAPACITAÇÃO EM SISTEMAS DE COMPUTAÇÃO - RECYCLING IS BETTER	
Líbero Passador Neto Dimitre Moreira Ort	
 <a href="https://doi.org/10.22533/at.ed.87021180816">https://doi.org/10.22533/at.ed.87021180816</a>	

<b>CAPÍTULO 17</b> .....	<b>206</b>
DESENVOLVIMENTO DE UM JOGO DIGITAL (2D) PARA CONSCIENTIZAÇÃO DE CRIANÇAS CONTRA O BULLYING	
Rafael Guedes da Silva Anderson Fabian Melo Nakanome	
 <a href="https://doi.org/10.22533/at.ed.87021180817">https://doi.org/10.22533/at.ed.87021180817</a>	
<b>CAPÍTULO 18</b> .....	<b>215</b>
DESENVOLVIMENTO DE UMA APLICAÇÃO WEB PARA PROPORÇÃO E DIMENSIONAMENTO DE AMOSTRA POR MEIO DO PACOTE SHINY	
Pablo Fellipe de Souza Almeida Cristina Henriques Nogueira	
 <a href="https://doi.org/10.22533/at.ed.87021180818">https://doi.org/10.22533/at.ed.87021180818</a>	
<b>CAPÍTULO 19</b> .....	<b>226</b>
DESIGN PATTERNS FOR SOFTWARE EVOLUTION REQUIREMENTS	
Anna Grimán Padua Manuel Capel Tuñón Eladio Garví	
 <a href="https://doi.org/10.22533/at.ed.87021180819">https://doi.org/10.22533/at.ed.87021180819</a>	
<b>CAPÍTULO 20</b> .....	<b>240</b>
DETECTION AND SEGMENTATION OF PIG IRON SLAG SCRAPERS USING MASK RCNN FOR WEAR CONTROL	
Carlos Eduardo Oliveira Milanez Marco Antonio de Souza Leite Cuadros Gustavo Maia de Almeida	
 <a href="https://doi.org/10.22533/at.ed.87021180820">https://doi.org/10.22533/at.ed.87021180820</a>	
<b>CAPÍTULO 21</b> .....	<b>252</b>
DIMENSIONAMENTO DE BLOCOS SOBRE ESTACAS METÁLICAS	
Fernanda Calado Mendonça Bernardo Horowitz	
 <a href="https://doi.org/10.22533/at.ed.87021180821">https://doi.org/10.22533/at.ed.87021180821</a>	
<b>CAPÍTULO 22</b> .....	<b>268</b>
ESTIMATION OF STELLAR PARAMETERS FOR J-PLUS SURVEY WITH MACHINE LEARNING	
Carlos Andres Galarza Arevalo Simone Daflon Vinicius Moris Placco Carlos Allende-Prieto	
 <a href="https://doi.org/10.22533/at.ed.87021180822">https://doi.org/10.22533/at.ed.87021180822</a>	
<b>CAPÍTULO 23</b> .....	<b>279</b>
ESTUDO ANALÍTICO E NUMÉRICO VIA MÉTODO DOS ELEMENTOS FINITOS DA	

## RIGIDEZ DOS PILARES DE PONTES EM CONCRETO ARMADO

Sávio Torres Melo  
Rebeka Manuela Lobo Sousa  
Pablo Juan Lopes e Silva Santos  
Francisca Itaynara de Souza Araújo  
Thiago Rodrigues Piauilino Ribeiro  
Amanda Evelyn Barbosa de Aquino  
Diogo Raniere Ramos e Silva  
Tiago Monteiro de Carvalho  
Carlos Henrique Leal Viana  
João Paulo dos Santos Silva  
Madson Nogueira da Silva  
Ilanna Castelo Branco Mesquita

 <https://doi.org/10.22533/at.ed.87021180823>

## **CAPÍTULO 24..... 290**

### **ESTUDO ANALÍTICO E NUMÉRICO VIA MÉTODO DOS ELEMENTOS FINITOS DOS EFEITOS DE SEGUNDA ORDEM EM PILARES DE PONTES EM CONCRETO ARMADO**


Sávio Torres Melo  
Rebeka Manuela Lobo Sousa  
Pablo Juan Lopes e Silva Santos  
Francisca Itaynara de Souza Araújo  
Thiago Rodrigues Piauilino Ribeiro  
Amanda Evelyn Barbosa de Aquino  
Diogo Raniere Ramos e Silva  
Tiago Monteiro de Carvalho  
Carlos Henrique Leal Viana  
João Paulo dos Santos Silva  
Madson Nogueira da Silva  
Ilanna Castelo Branco Mesquita

 <https://doi.org/10.22533/at.ed.87021180824>

## **CAPÍTULO 25..... 311**

### **ESTUDO DO MOVIMENTO DOS CORPOS MOEDORES NO PROCESSO DE MOAGEM UTILIZANDO O MÉTODO DOS ELEMENTOS DISCRETOS**


Wladimir José Gomes Florêncio  
Neilor Cesar dos Santos



 <https://doi.org/10.22533/at.ed.87021180825>

## **CAPÍTULO 26..... 329**

### **FLUID FLOW SUMMARIZATION USING DYNAMIC MULTI-VECTOR FEATURE SPACES**

Renato José Policani Borseti  
Leandro Tavares da Silva  
Gilson Antonio Giralaldi

 <https://doi.org/10.22533/at.ed.87021180826>

<b>CAPÍTULO 27</b> .....	<b>351</b>
GESTÃO DE PROCESSOS: ALINHAMENTO ESTRATÉGICO ENTRE TI E NEGÓCIO COM BPMN	
Aryel Evelin Vieira Garcia Rodrigo Elias Francisco	
 <a href="https://doi.org/10.22533/at.ed.87021180827">https://doi.org/10.22533/at.ed.87021180827</a>	
<b>CAPÍTULO 28</b> .....	<b>359</b>
IFINTEGRA - INTEGRADOR DA PLATAFORMA DE REGISTRO DE PROJETOS COM A BIBLIOTECA DIGITAL DE ARTIGOS DE UM CAMPUS DO IFSUL	
Mateus Roberto Algayer Geovane Griesang	
 <a href="https://doi.org/10.22533/at.ed.87021180828">https://doi.org/10.22533/at.ed.87021180828</a>	
<b>SOBRE O ORGANIZADOR</b> .....	<b>366</b>
<b>ÍNDICE REMISSIVO</b> .....	<b>367</b>



## DESIGN PATTERNS FOR SOFTWARE EVOLUTION REQUIREMENTS

*Data de aceite: 02/08/2021*

*Data de submissão: 06/07/2021*

### **Anna Grimán Padua**

Process and Systems Department, Simón Bolívar University  
Caracas - Venezuela  
<https://orcid.org/0000-0002-9092-6952>

### **Manuel Capel Tuñón**

Software Engineering Department, College of Informatics and Telecommunications, University of Granada  
Granada - Spain  
<https://orcid.org/0000-0003-2449-4394>

### **Eladio Garví**

Software Engineering Department, College of Informatics and Telecommunications, University of Granada  
Granada - Spain  
<https://orcid.org/0000-0002-9267-1711>

**ABSTRACT:** The Software Engineering term known as Software Evolution can be understood in two senses. First, as the changes experienced by software over the development cycle; second, software changes while the system is in service. In both cases, software architectures should lead, support and ease any modifications, reconfiguration or adaptation to a changing environment, of software systems. We present here the analysis of several design and architectural patterns for sustaining software systems evolution according to two perspectives:

one connected with maintainability and the other with dynamicity of changes of any software design. We also study weaknesses and benefits, aligned with ISO 25010 standard characteristics, of each selected pattern in such way that they can easily be analyzed as candidate in a context of architectural improvement. At moment, it is widely acknowledged that design and architectural patterns should be used for software development focused on quality. The analysis conducted on patterns facilitates software evolution by easing software changes throughout software development phases and further maintenance of the final product.

**KEYWORDS:** Maintainability, Architectural Patterns, Design Patterns, Software Quality Standard, Dynamic Reconfiguration.

### **PADRÕES DE DESIGN PARA REQUISITOS DE EVOLUÇÃO DE SOFTWARE**

**RESUMO:** O termo engenharia de software conhecido como Evolução de Software pode ser entendido em dois sentidos. Primeiro, como as mudanças experimentadas pelo software ao longo do ciclo de desenvolvimento; segundo, o software muda enquanto o sistema está em serviço. Em ambos os casos, as arquiteturas de software devem liderar, apoiar e facilitar quaisquer modificações, reconfiguração ou adaptação a um ambiente em mudança, de sistemas de software. Apresentamos aqui a análise de diversos padrões de design e arquitetônicos para sustentar a evolução dos sistemas de software de acordo com duas perspectivas, uma conectada à

manutenção e outra com dinâmica de mudanças de qualquer design de software. Também estudamos fraquezas e benefícios, alinhados com as características padrão ISO 25010, de cada padrão selecionado de forma que possam ser facilmente analisados como candidatos em um contexto de melhoria arquitetônica. No momento, é amplamente reconhecido que o design e os padrões arquitetônicos devem ser usados para o desenvolvimento de software focado na qualidade. A análise conduzida em padrões facilita a evolução do software, facilitando as mudanças de software ao longo das fases de desenvolvimento de software e posterior manutenção do produto final.

**PALAVRAS-CHAVE:** Manutenção de software, Padrões Arquitetônicos, Padrões de Design, Padrão de Qualidade de Software, Reconfiguração Dinâmica.

## 1 | INTRODUCTION

Over the last two decades and after publishing seminal works as Bass et al (2003), Software Architecture (SA) has increasingly created interest. Researchers and ICT professionals have come to acknowledge the paramount importance of SA in software's quality characteristics (Boch J., 2000; Grimán et al., 2006; Grimán et al., 2006b). On the other part, some selected methods and techniques have been defined to early assess such characteristics through an architectural view (Kazman et al., 1996).

Design Patterns, Architectural Styles and Patterns are particular cases of architectural mechanisms to be considered in software development. Firstly proposed in the seminal work conducted by Gamma et al. (1996), these architectural mechanisms have experienced a great use since then due to Object-Oriented Programming and concurrent and distributed systems dissemination. Some studies have been carried out until now to assess the effect of design patterns to software quality, mainly through empirical methods (Gommathy, 2008; Harrison & Avgeriou, 2007; Bode & Riebish, 2010; Ozkaya et al, 2007), but their conclusions with regard to quality characteristics show different and controversial results. Moreover, we only found few articles in the literature of SA regarding the importance of design patterns for successful software evolution (Lung et al, 1997).

The main objective of this paper is an initial proposal for reviewing the most popular GoF design patterns to determine their usefulness in evolutionary software and dynamic architectures. Our baseline here is to analyze each pattern, within a set of great use in industry nowadays, regarding quality characteristics according to ISO 25010 standard (ISO/IEC, 2011).

Since our study is deeply based on design and architectural patterns, we needed to identify a significant subset of GoF patterns, which analysis could give us a good foundation to predict quality characteristics of many software applications that normally make use of these patterns.

Section 2 is devoted to introduce some previous background on the impact that patterns have in the quality of evolutive software and then, in Section 3, the selection of the set of patterns for our study is explained.

Then, we will present our approach and show that is capable of identifying the important benefits and “sensibility points” that can propitiate some software quality characteristics and inhibit others at the same time.

In Section 4 a set of Design Patterns are presented in order to analyze their main properties and to conclude different degrees of maintainability for software systems that use each one of these patterns. Section 5 presents a second set of patterns, in this case architectural patterns, in order to study their distinct support degree of dynamic software architecture reconfiguration. Finally, the future work and conclusions are presented in Section 6.

## **2 | BACKGROUND**

The basic idea behind this research is that the desired quality of any software piece can be firstly attained by the level of concordance between selected architectural decisions and the functional and supplementary requirements of the system being modeled. Among the diverse architectural decisions that software architects can make, the usage of certain design patterns and styles is of interest for this investigation because these are capable of providing software quality.

On other hand, we will name evolutionary software the software that goes through different versions over the development process. Different versions can be also understood as successive versions of the same software, which perform changes correspondent to the product specification either by adding or removing requirements. The latter fact brings into focus the necessity of establishing the quality characteristics associated to software evolution. In order to achieve this goal, we will use the acknowledged ISO 25010 quality standard.

Our investigation is based on a previous study conducted by Grimán et al (2006), that we have adapted to the alignments established by the standard above.

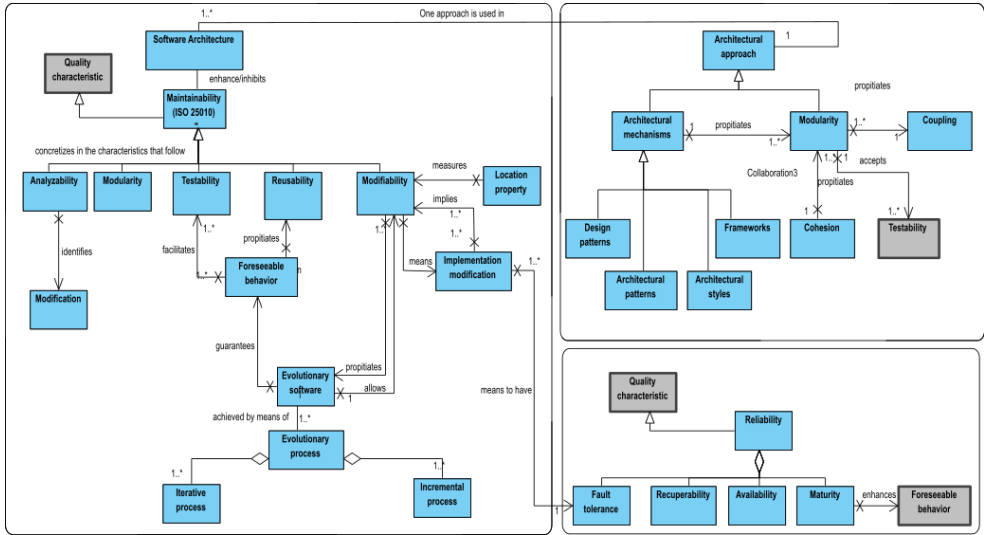


Fig. 1. Ontology for Maintainability as defined by ISO 25010 Model. Adapted from Grimán et al (2006)

In figure 1 we can see the ISO 25010based ontology of “evolutionary software”, where we can observe that evolutionary requirements of software can be expressed in terms of *Maintainability* and *Reliability* characteristics. This ontology allow us to establish metrics, which are inspired on the subcharacteristics of software quality related to software evolution and disclosed in the contributed work (Grimán et al, 2006).

In order to analyze the level of concordance between architectural decisions and system requirements, we firstly need to establish the quality characteristics associated to the evolution of a software product. In order to achieve that objective, we use the acknowledged ISO 25010 quality standard quality (ISO/IEC, 2011).

We needed a set of patterns from which we can derive the quality attributes that each one enhances or inhibits, then we will translate these into ISO 25010 quality characteristics. In doing so, we will need to follow a critical analysis process in order to assess quality characteristics of each selected pattern: a) to select a relevant set of design and architectural patterns; b) to identify the Maintainability subcharacteristics in which we want to focus on our valuation; c) to determine which design or architectural patterns can enhance and which ones can punish this subcharacteristic; and d) to recommend the usage of the well-assessed patterns.

### 3 | PATTERNS SELECTION

We were more interested in patterns whose scope will be objects, not a class, since the software that we are interested to maintain and make evolve is based on complex structures that need collaborative relations between objects. Among the set of 23 GoF patterns (Gamma et al, 1996) in Table 1, we selected the emphasized ones (blue) to

conduct our study.

By purpose		Creational	Structural	Behavioral
By scope	Class	Factory method	Adapter (class)	Interpreter Template method
	Object	Abstract factory Builder Prototype Singleton	Adapter (object) Bridge Composite Decorator Façade Flyweight Proxy	Chain of responsibility Command Iterator Mediator Memento Observer State Strategy Visitor

Table 1. GoF Patterns (Gamma et al, 1996).

In current applications, objects get created and deleted dynamically in accordance with the real world, which is plenty of ubiquitous devices filled up with business logic and intelligence. *Factory method*, *Singleton*, *Adapter (class)*, *Interpreter* and *Template method* are therefore not selected for analysis of quality characteristics either.

Secondly, Maintainability of any software architecture is transparent of those design patterns that only temporally store the state of an object or a software application (*Memento*), or those that only traverse a structure of objects (*Iterator*) or make easier representation and traversal of complex object hierarchies by recursive searches like DFS (*Composite*).

We did not include *Abstract Factory* (AF) design pattern as it leads to an appreciable code redundancy, since this pattern induces duplication of code and then causes further problems to build and later maintenance of software. `FACTORY`-classes and their descendants need to be changed if new products are needed in the catalogue, thus AF pattern shows lack of flexibility with respect to new inclusions of concrete classes' objects. Moreover, any future changes to be performed, or any error or fault correction that might arise, will impact to all the variants of the same code. As consequence, the use of AF turns the code complex and difficult to analyze or change. Thus, we decided not to assess the patterns: *Abstract Factory*, nor other creational patterns like *Builder*, *Prototype* and *Singleton* as a first approach of our study.

In our experience, when using *Observer* pattern error traceability becomes difficult if not impossible to carry out. This may seriously inhibit the attributes of Modifiability and Testability of any software. Thus, we can affirm that Testability and Recoverability sub-characteristic are affected. Testing any software that uses *Observer* with guarantee becomes quite complex. Thus, it is not considered for analysis in the first part of the study presented here.

On the other hand, patterns that encapsulate application's behavior, at an execution

instant, by storing the internal state of the application objects make possible to change the behavior at run time in a flexible way (*State, Strategy*) without resorting to unnecessarily populate the code with conditional statements. The pattern *Strategy* therefore support dynamic reconfiguration at run time and greatly improve the maintainability of any software and thus.

In Sections 4-5, we present in a more systematic way the results of the analysis of the whole set of selected GoF design patterns.

Dynamic Architectural Reconfiguration is an important evolutionary aspect in software development nowadays, especially, for the requirements of self-adaptability, multicore load balancing and resilience to failures that collaborative applications present today.

The following patterns allow us to run time architecture reconfiguration: *Decorator, Strategy, Broker, and Bridge*. Additionally, we found some architectural patterns, which support a dynamic reconfiguration from a higher level of abstraction (Schmidt et al, 1999). We decided to incorporate a subset of these patterns with the purpose of observing changes in the maintainability metrics responses when introducing high-level architectural decisions. We selected the following patterns: *Component Configurator (CC), Interceptor, Extension Interface, and Reflection*. Thus, we conducted a theoretical and critical analysis of these 8 patterns in order to better complete our evaluation of patterns for evolutionary software.

#### 4 I PATTERNS PROPITIATING HIGHER MAINTAINABILITY LEVELS

In this section, we present the analysis of a subset of the selected patterns. This first approach included 7 design patterns. Since this kind of patterns provide very detail information, we were able to analyze the impact of different low-level architectural decisions. A theoretical and critical analysis that includes a description of every pattern, their components and application, as well as the pros and cons of their use in terms of the quality characteristics according to ISO 25010 standard (ISO/IEC, 2011) has been carried in our study (see Tables 2-3). For each pattern and ISO characteristic, we identify whether a particular sub-characteristic of the produced software gets (*Impact*) enhanced (+) or inhibited (). Characteristic assessment has been carried by application of appropriate metrics based on the Maintainability ontology described in Figure 1 (Section 2).

For space reasons, following we briefly present the analysis for a single design pattern: **Wrapper-façade (WF)** pattern.

WF encapsulates functions and data provided by nonobject oriented existing APIs inside more concise, robust, portable, easytomaintain and cohesive interfaces.

Our aim when using this pattern is to avoid the deployment of conditional compilation directives and direct programming of APIs that make the code difficult to understand, debug, transfer and evolve. By the creation of a class named Wf that encapsulates spare functions and data throughout nonobject oriented coded APIs, conditional compilation, scripts and

other inclusions in the object-oriented base code become unnecessary. Thus, the methods defined in a WF interface are called instead of macro expansion.

After analyzing these properties, in Table 2 we can see a compilation of pattern's attributes and ISO quality characteristics that seem to get enhanced or inhibited if we use the pattern WF to design one part of any software. Notice that by using the pattern WF, the following quality characteristics and subcharacteristics were enhanced in the final software product: Maintainability/ modifiability, modularity, reusability; Performance/ resource usage; Compatibility/ coexistence, interoperability; Usability/ appropriateness, learnability, user error protection, user interface aesthetics; Functional suitability/ completeness, correctness, appropriateness.

Characteristic	Sub-characteristic	Impact	Metric
<b>Maintainability</b>	Testability	-	Dynamicity
	Modifiability	+	Cohesiveness
	Reusability	+	Reusable components existence
	Modularity	+	Cohesiveness Complexity (low)
	Analysability	+	Inheritance relationships and generic types
<b>Performance</b>	Response time	-	Contextual consistency
	Resource usage	+	Encapsulation
<b>Portability</b>	Flexibility	-	Compatibility
	Adaptability	-	Environment dependencies
<b>Security</b>	Integrity (*)	+	Cohesiveness
		-	Contextual consistency
<b>Reliability</b>	Maturity	+	Encapsulation
		+	Complexity
		+	Cohesiveness
		-	Conformance to standards
		-	External dependencies
<b>Compatibility</b>	Co-existence	+	Coupling (low)
	Interoperability	+	Unified interfaces
<b>Usability</b>	Appropriateness	+	Simplicity
	Learnability	+	Complexity (low)
	Operability	-	Deployability
	User error protection	+	Cohesiveness of interfaces
	User interface aesthetics	+	Avoids APIs direct programming
	Accessibility	-	Flexibility

<b>Functional suitability</b>	Completeness	+	Analyzability
	Correctness	+	Compositionality of interfaces
	Appropriateness	+	Traceability

Table 2. Quality characteristics of Wrapper-Façade design pattern.

As a conclusion of the analysis presented in Table 2, we can affirm that software maintainability gets enhanced because the WF pattern creates cohesive and reusable components that can be interconnected inside container-components through inheritance relationships and generic types. Moreover, encapsulation and cohesiveness propitiate application stability since changes are mainly bounded to component interfaces and not to component implementation.

We can identify (Table 2) several ISO 25010 sub-characteristics to be connected to pattern WF. The majority of them are (complete or partially) enhanced by pattern WF and 6 of them get completely inhibited, i.e., Testability, Accessibility, Operability, Flexibility, Adaptability and Response Time.

Following, Table 3 summarizes the analysis results for the rest of the GoF selected patterns. Notice that those patterns promoting dynamic reconfiguration of software were included as a part of the analysis presented in section 5.

Pattern	Characteristic	Sub-characteristic	Impact	Metric
<b>Adapter</b>	Maintainability	Modifiability	+ -	Functional extension Interfaces comprehension and integrity
		Reusability	+	Reusable components
		Analysability	- -	Complexity Unexpected changes
	Performance Efficiency	Response time	+	Deployment
		Resource usage	+	Class hierarchies reduction
	Security	Integrity	+ -	Flexibility Reconfiguration
	Reliability	Fault tolerance	- -	Adaptor's inheritance Difficult error handling of ADAPTEES
		Maturity	+ +	Calls handling Encapsulation



<b>Chain of Respon.</b>	Maintainability	Modifiability	+	Unified interfaces	
		Reusability	+	Low coupling	
		Analysability	-	Dynamicity	
	Performance	Response time	-	Response predictability	
		Resource usage	+	Encapsulation	
	Security	Integrity	+ -	Flexibility Reconfiguration	
	Reliability	Fault tolerance	- +	Lost calls Scalability	
		Maturity	+ +	Coupling Accuracy	
	<b>Mediator</b>	Maintainability	Modifiability	+	Encapsulation
			Reusability	+	Cohesiveness
Analysability			+	Coupling (low)	
Testability			-	Complexity (increasing)	
Performance		Resource usage	+	Encapsulation	
Portability		Interoperability	+	Contextual independence	
Security		Integrity	+	Consistency (internal, external)	
<b>Visitor</b>	Maintainability	Modifiability	+	Encapsulation	
		Analysability	+	Dynamicity	
		Testability	+	Occurred events monitoring	
	Security	Integrity	+	Consistency	
	Performance	Resource usage	+	Abstract visiting draws code economy	
		Response time	-	Class hierarchies extension	
	Reliability	Fault tolerance	+	Error handling centralization	
	Portability	Adaptability	+	Interoperability	
<b>Facade</b>	Maintainability	Modifiability	+	Simplicity	
		Analysability	+	Interfaces unification	
		Testability	+	Encapsulation	
	Portability	Adaptability	+	Interoperability	
	Performance	Resource usage	+	Coupling (low)	
		Response time	+	Simpler communications	
	Security	Integrity	+	Encapsulation	
	Reliability	Maturity	+	Encapsulation	
<b>Command</b>	Maintainability	Modifiability	+	Unified interfaces	
		Analysability	+	Auditing (logs)	
			+	Simplicity	
		Testability	+	Traceability	
	Performance	Resource usage	+	Encapsulation	

Table 3. Quality characteristics associated to the selected design patterns.

Based on the analysis presented in Table 3, we can affirm that software maintainability is total or partially supported by the whole set of selected patterns. At the same time, the following quality characteristics are promoted by these patterns: Performance, Security, Reliability, and Portability. On the other hand, we can observe in Table 3 that certain sub-characteristics (such as Analyzability and Fault Tolerance) got completely inhibited because of the usage of Adapter pattern.

## 5 I PATTERNS FOR DYNAMIC ARCHITECTURES

As a rationale for Dynamic Configuration analysis we assessed those services provided on the server side, such as scheduling mechanisms for clients and servers.

For the study of this particular requirement of software evolution we included a different subset of the patterns selected in section 3 that facilitate run time architecture reconfiguration: *Decorator*, *Strategy*, *Broker*, *Bridge*, *Component Configurator*, *Interceptor*, *Extension Interface*, and *Reflection*. Since this subset includes both design and architectural patterns, we were able to analyze the impact of architectural decisions presenting different levels of abstraction. The fundamental pros and cons of these patterns regarding ISO 25010 characteristics are shown in Table 4. For space reasons, we only present a brief analysis of **Extension Interface** pattern.

This pattern aims at multiple interfaces exportation by only one component, so that modification or extension of a component functionality will not break the source client code or prevent the definition of interfaces with a high level of complexity.

The fundamental insight we must obtain from this pattern is separation of concerns when programming clients so that they can access components through separate interfaces, one interface per component role instead of mixing all the roles in the same interface or component implementation. Even though *Extension Interface* is an effective mechanism, application timeliness can be compromised because it is complex and prone to experience locks. The interesting characteristic of this pattern lies in the use of interfaces and the implementation of abstract methods more than the sub-classing mechanism. Thus, development of factorybased software components is promoted with this pattern.

Pattern	Characteristic	Sub-characteristic	Impact	Metric
<b>Comp. Configurator</b>	Maintainability	Testability	+	Independence
		Modifiability	+	Dynamicity
		Reusability	+	Cohesiveness
		Modularity	+	Functional extension
		Analysability	-	Comprehensibility
	Performance	Response time	-	Complexity
		Resource usage	+	Low coupling
	Portability	Adaptability	-	Standardization
	Security	Integrity	-	Vulnerability
	Reliability	Maturity	-	Integration
<b>Interceptor</b>	Maintainability	Modifiability	+	Encapsulation
		Reusability	+	Extensibility
			+	Flexibility
		+	Coupling	
		+	Dynamicity	
		+	Complexity	
	Analysability	-	Extensibility	
		-	Flexibility	
	-	Complexity		
Performance	Response time	-	Complexity	
	Resource usage	+	Monitoring control	
Reliability	Fault tolerance	-	Defective timeliness	
		+	Monitoring control	
	Maturity	-	Defective timeliness	
<b>Ext. Interface</b>	Maintainability	Modifiability	+	Cohesion
		Reusability	+	Polymorphism
			+	Coupling
		+	Dynamicity	
	Performance	Response time	-	Complexity
		Resource usage	-	Complexity
	Compatibility	Interoperability	-	Complexity
			+	Extensibility
	Reliability	Fault tolerance	-	Locks
			+	Monitoring control
Maturity		-	Timeliness lock	
<b>Decorator</b>	Maintainability	Modifiability	+	Flexibility
			+	Extensibility
			+	Dynamicity
	Analysability	+	Flexibility	
		-	Complexity	
Testability	-	Complexity		
Performance	Resource usage	+	Complexity	

<b>Strategy</b>	Maintainability	Modifiability	+	Extensibility
		Reusability	+ +	Simplicity Dynamicity
		Analysability	-	Complexity
		Testability	+ +	Extensibility Simplicity
	Performance	Resource usage	-	Complexity
	Functional suitability	Completeness	+	Extensibility
	Compatibility	Interoperability	-	Complexity
<b>Broker</b>	Maintainability	Modifiability	+	Flexibility
		Analysability	-	Simplicity
		Testability	-	Hiding
	Compatibility	Interoperability	-	Transparency
	Performance	Resource usage	-	Optimization
		Capacity	+	Scalability
		Response time	-	Complexity
Reliability	Fault tolerance	-	Hiding	
Portability	Adaptability	+	Multi-platform	
<b>Reflection</b>	Maintainability	Modifiability	+ +	Extensibility Dynamicity
		Analysability	-	Complexity
		Testability	-	Complexity
	Functional suitability	Correctness	+	Accuracy
	Performance	Resource usage	-	Complexity
	Reliability	Maturity	-	Complexity
		Fault tolerance	-	Failure biased

Table 4. Quality characteristics of the selected dynamic patterns.

From the analysis presented in Table 4, we can affirm that dynamic architecture partially promoted by the whole set of selected patterns. Notice that the sub-characteristic Analyzability got inhibited in all cases. On the other hand, the following quality characteristics are partially promoted by these patterns: Performance, Reliability, Compatibility, Functional Suitability and Portability.

## 6 | CONCLUSIONS AND FUTURE WORK

The analyzed patterns benefit software evolution from two aspects, according to discussed approaches in this paper, Maintainability and Dynamicity. Worth to mention is the variety of different mechanisms, common to design and architectural patterns, such as coupling, encapsulation, inheritance, and complexity, among others, which are used to

guarantee such aspects. We also observed in our analysis that these characteristics and subcharacteristics are rarely achieved without punishing others, like Performance Efficiency or Reliability. However, according to the results obtained in this research, we can affirm that the usage of patterns is an effective way for constructing software if evolution requirements have to be satisfied. We recommend, during the software design phase, to incorporate design and architectural patterns within a rigorous analysis that considers the benefits and sensibility points (tradeoffs) exposed here.

We can see that these presented patterns are not excluding alternatives, but it is advisable to combine several of them, thus causing a greater heterogeneity in the target software architecture and then it makes sense to perform architectural assessments that take into account the effect of applying a set of these patterns.

Finally, we recognize the limitations of the theoretical approach followed in this research; however, the obtained results allow us to establish a preliminary set of hypothesis for an empirical evaluation of the affected characteristics and sub-characteristics. Currently, we are working in the preparation of different experiments within academic as well as industrial settings to implement and measure these results using Java applications.

## REFERENCES

BASS, L. *et al.* **Software Architecture in Practice**. 1. ed. Addison-Wesley, 2003.

BODE, S.; RIEBISH, M. Impact evaluation for quality oriented architectural decisions regarding evolvability. Lecture Notes on Computer Science, 6285, pp. 182-197, 2010. Article presented in ECSA 2010, Copenhagen, Denmark.

BOSCH, J. **Design and Use of Software Architecture**. England: ACM Press, 2000.

GAMMA, E. *et al.* **Design Patterns: elements of reusable object oriented software**. Addison-Wesley, 1994.

GOMATHY, C.K. Software pattern quality compartment in service-oriented architectures. **European Scientific Journal**, v.10, n. 92014, March 2014, pp.:412-423.

GRIMAN, A. *et al.* Towards a Maintainability Evaluation in Software Architectures. *In: 8th International Conference on Enterprise Information Systems ICEIS*, 2006.

GRIMAN, A *et al.* Characteristic analysis for architectural evaluation methods. **Journal of Systems and Software**, v. 79, n. 6, 2006, pp.: 871-888.

HARRISON N.; AVGERIOU P. Leveraging architecture patterns to satisfy quality attributes. Lecture Notes on Computer Science, 4758, pp. 263-270, 2007. Article presented in ECSA 2007, Aranjuez, Spain.

ISO - International Organization for Standardization. **ISO/IEC 25010:2011 (en)**: Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models. Geneva: ISO, 2011.

KAZMAN, R. *et al.* ScenarioBased Analysis of Software Architecture, **IEEE Software**, V. 13, n.6, Nov. 1996.

LUNG, C.H. *et al.* An Approach to Software Architecture Analysis for Evolution and Reusability. *In: SEI Digital Library*. USA, 1996. Available on: <https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=29765>. Retrieved on: 12 jan 2021.

OZKAYA, I. *et al.* **QualityAttributeBased Economic Valuation of Architectural Patterns**. 2007. Software Engineering Institute. DOI 10.1184/R1/6582686.v1. Available on: <https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=8307>. Retrieved on: 12 jan 2021.

SCHMIDT, D. C. *et al.* **Patterns for Concurrent and Networked Objects**. Chichester, UK: Wiley, 2000.

## ÍNDICE REMISSIVO

### A

Algoritmo 38, 40, 42, 44, 46, 48, 50, 70, 82, 120, 168, 169, 182, 257, 262, 265, 322, 330

Análise avançada 53, 54, 55, 68

Análise computacional 84, 103

Análise estrutural 55, 71, 82, 84, 85, 92, 93, 94, 95, 97, 103, 109, 110, 111

Aprendizado 13, 174, 193, 194, 197, 208, 215, 224, 268

### B

Bullying 206, 207, 208, 210, 211, 212, 213, 214

### C

Carga crítica 143, 144, 147, 148, 149, 152, 153

Computational fluid dynamics 329, 330, 350

Constitutive model 1, 2, 5, 6, 10

Contorno 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 49, 50, 51, 71, 299

Controlador neural 168, 169, 170, 171, 172, 173, 174, 175, 179

Controle 19, 119, 120, 131, 168, 169, 171, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 240, 295, 312, 352, 353, 356, 358

### D

Deep learning 131, 132, 134, 135, 136, 137, 138, 141, 142

Descarte adequado 198

Desenvolvimento 11, 12, 14, 15, 17, 20, 21, 35, 36, 40, 44, 82, 83, 193, 194, 195, 197, 198, 199, 200, 205, 206, 208, 209, 210, 211, 215, 216, 217, 221, 225, 226, 227, 254, 260, 265, 281, 294, 327, 352, 354, 357, 359, 361, 362, 363, 364

Design patterns 155, 156, 166, 167, 226, 227, 228, 230, 231, 234, 238

Diferenças finitas 38, 39, 40, 45, 50, 51, 52, 315

Digital 167, 197, 206, 207, 210, 213, 239, 243, 319, 320, 358, 359, 360, 362, 363, 365

Drop test 131, 132, 133, 134, 135, 141

### E

Educação 12, 13, 14, 21, 53, 68, 70, 191, 193, 195, 197, 208, 212, 215, 225, 279, 290, 311, 326, 359, 366

Educacional 14, 82, 206, 208, 209

Elemento hexaédrico 70, 72, 75, 77

Elementos finitos 53, 55, 69, 70, 71, 72, 83, 279, 280, 281, 285, 286, 290, 291, 294, 297,

299, 303, 306, 309, 321

Equações diferenciais 39, 40, 44, 51, 71, 294

Estabilidade estrutural 143

Estatística 21, 215, 216, 217, 218, 224, 225

Estrutura 17, 38, 54, 71, 72, 75, 77, 78, 81, 82, 84, 85, 87, 89, 90, 91, 97, 98, 99, 101, 102, 103, 104, 106, 112, 113, 114, 115, 116, 117, 118, 152, 218, 221, 253, 266, 279, 280, 281, 282, 283, 284, 286, 288, 291, 292, 293, 297, 298, 302, 309, 362, 363

## **F**

Ferramenta 15, 18, 22, 39, 193, 194, 195, 196, 200, 204, 210, 211, 216, 224, 294, 313, 354, 356, 360, 361, 363

Frequências naturais 143, 144, 146, 147, 149, 150, 151, 152, 153

Fundação elástica 143

## **G**

Geometria irregular 38

Gestão de processos 351, 352, 354, 355, 358

## **I**

Imperfeições geométricas iniciais 53, 54, 55, 62, 64, 67, 69

Inclusão 29, 33, 35, 36, 67, 68, 197, 359, 360

Industrial process 131

Informação 12, 21, 193, 205, 216, 351, 354, 355, 356, 357, 358, 360, 366

Inovação 86, 104, 105, 193, 366

Interfaces 215, 216, 225, 231, 232, 233, 234, 235, 361

## **J**

Jogo 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 206, 207, 209, 210, 211, 212, 213

## **L**

Layout 221, 222, 359, 360, 362

Libras 11, 12, 13, 14, 17, 18, 19, 20, 21, 22

## **M**

Malha 38, 39, 40, 44, 45, 46, 49, 50, 72, 79, 108, 182, 285, 299, 303, 304, 313, 321, 322, 326

Modelagem 31, 33, 35, 36, 38, 39, 70, 72, 149, 194, 251, 255, 268, 280, 285, 294, 295, 299, 305, 351, 352, 353, 354, 356, 357, 358

Modos incompatíveis 70, 72, 75, 76, 77, 79, 80, 82, 83



## O

Oscar Niemeyer 84, 85, 86, 87, 89, 101, 102, 103, 104, 105, 118

## P

Pasternak 143, 144, 145, 149, 151, 153, 154

Processos 82, 171, 240, 312, 351, 352, 353, 354, 355, 356, 357, 358, 359, 361

Programação 72, 211, 215, 224, 361

Programas 55, 205, 206, 210, 214, 294, 359

Projeto socioambiental 198

## R

Realidade aumentada 193, 194, 195, 196, 197

Rede neural 168, 169, 171, 175

Resistência 53, 54, 55, 56, 60, 61, 62, 63, 64, 65, 66, 67, 68, 89, 96, 131, 145, 255, 256, 258, 261, 262, 263, 280, 294, 314

Robô 168, 169, 170, 173, 174, 175, 176, 177, 178, 179

Robótica 168

RPG 11, 12, 15, 16, 18

RStudio 215, 216, 217, 218, 220, 224, 225

## S

Shiny 215, 216, 217, 218, 220, 221, 224, 225

Simulações 23, 24, 30, 31, 33, 35, 38, 44, 50, 168, 169, 175, 181, 311, 312, 326, 329

Sobretensões de manobras 23, 24, 25, 29, 30

Software 1, 6, 12, 18, 40, 53, 55, 66, 70, 71, 72, 77, 79, 80, 82, 103, 155, 156, 157, 158, 159, 166, 167, 196, 210, 215, 216, 217, 218, 220, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 235, 237, 238, 239, 256, 257, 263, 265, 266, 267, 281, 285, 297, 299, 300, 311, 320, 321, 330, 356, 357, 359, 360, 363, 364

Stable hysteresis cycle 1, 3, 9

Summarization 329, 330, 331, 332, 343, 349, 350

Supressores de surto 23, 25, 28, 29, 30, 33, 34, 35, 36

Sustentabilidade 198, 199

## T

Tecnologia 11, 12, 21, 54, 70, 168, 193, 194, 196, 197, 206, 208, 215, 279, 290, 311, 326, 351, 355, 358, 359, 362, 366

Tensão 1, 24, 25, 26, 27, 28, 33, 34, 59, 62, 63, 66, 67, 75, 170, 255, 256, 258, 260, 261,

266, 295

Tensões residuais 53, 54, 55, 61, 62, 63, 64, 65, 66, 67, 68, 69

Transformadores 23, 24, 25, 28, 30, 34, 35, 36

Transitórios eletromagnéticos 23, 24, 31

## **W**

Web 54, 194, 195, 196, 200, 215, 216, 217, 218, 221, 222, 225, 355, 359, 360, 361, 362, 363, 365

COLEÇÃO

# DESAFIOS DAS ENGENHARIAS:





ENGENHARIA DE COMPUTAÇÃO

-  [www.atenaeditora.com.br](http://www.atenaeditora.com.br)
-  [contato@atenaeditora.com.br](mailto:contato@atenaeditora.com.br)
-  [@atenaeditora](https://www.instagram.com/atenaeditora)
-  [www.facebook.com/atenaeditora.com.br](https://www.facebook.com/atenaeditora.com.br)

COLEÇÃO

# DESAFIOS DAS ENGENHARIAS:

ENGENHARIA DE COMPUTAÇÃO

-  [www.atenaeditora.com.br](http://www.atenaeditora.com.br)
-  [contato@atenaeditora.com.br](mailto:contato@atenaeditora.com.br)
-  [@atenaeditora](https://www.instagram.com/atenaeditora)
-  [www.facebook.com/atenaeditora.com.br](https://www.facebook.com/atenaeditora.com.br)