



About Author:

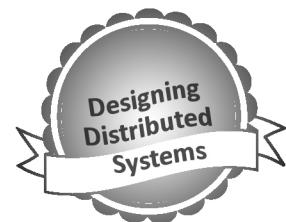
The Book '**Designing Distributed Systems**' is written by Cyber security expert Mr. Joseph Thachil George. Joseph writes books, which, considering where you're reading this, makes perfect sense. He is best known for writing research papers, including the technical and non- technical contents.

Joseph has taken bachelor's degree in Computer Science and Engineering from the Mahatma Gandhi University in Kerala, India, and he holds M.S in Cyber Security from the University of Florence, Italy. At present Joseph is working as a Game Developer and doing research in Cyber security and blockchain.

Joseph has seven years of experience in computer science and engineering field and he also done research projects for various Italian companies and governments such as IGT-Rome, Comune di Palermo, Italy, Confesercenti Rome

Content:

1. Introduction	5.	10. Model-Driven Engineering	78.
2. Introduction to Cyber-Physical Systems	6.	1. Designing a metamodel for CPSoS	84.
1. Emergence	8.	2. Blockly 4sos	87.
2. Systems of Systems	9.		
3. Managing Time	9.	11. Project using MATLAB-Smart-Farm	100.
4. Coordinated Clocks	10.		
5. Data and State	11.	12. System Implementation	101.
6. Actions and Behaviour	12.	1. Environment Models	101.
7. Communication	13.	2. Sensor Models	102.
8. Stigmergy	14.	3. Multi-Robot Lidar Sensor	102.
9. Interfaces	15.		
10. Evolution and Dynamicity	16.	13. System Architecture	102.
11. System Design and Tools	16.		
12. Dependability and Security	17.	14. System Modeling	104.
		1. Robot Visualizer and Lidar Sensor	104.
3. Interfaces in Evolving Cyber-Physical Systems of Systems (CPSoSs)	21.	2. Obstacle avoidance logic and 2PC protocol concept	105.
1. Interface Layers	23.	3. Architecture of North & South Farm and Store house	105.
2. Relied Upon Interface (RUI)	27.		
3. Handling Evolution at RUIs	31.	15. Implementation of Two-Phase Commit protocol	107.
4. Emergence in Cyber-Physical Systems of Systems (CPSoSs)	32.	16. Requirements	108.
1. Emergence	33.	1. Mutual exclusion, Safety	108.
2. Examples	39.	2. Liveness	108.
3. Consequences for System Design	41.	3. Fairness	108.
5. Distributed Coordination of Systems (CPSoSs)	43.	17. Problem Encountered	109.
1. Event Ordering	44.		
2. Mutual Exclusion	45.	18. Conclusion	110.
3. Atomicity	46.		
4. Concurrency Control	49.		
5. Deadlock Handling	50.		
6. Election Algorithms	51.		
6. Failure Detectors Model	53.		
7. Consensus Algorithms For Blockchains	59.		
8. Real-Time Systems	67.		
9. Scheduling in real-time Systems	73.		



LIST OF FIGURES

1. Overview of CPS	07
2. Emergent phenomena	08
3. Time Standard	10
4. Coordinated Clocks	10
5. Data and information	11
6. Sampling	12
7. Communication	13
8. Comparison of Messages	14
9. Stigmergy	14
10. Trail-pheromone of stigmergy	15
11. Channel of communication	15
12. Dependability Overview	18
13. Error	19
14. Dependability: Attributes	19
15. Dependability: Means	20
16. Overlapping entourage of CPSs enabling physical interaction of CPSs enabling physical interaction	23
17. Example: Emergency Braking	26
18. Interface Layers	26
19. Interfaces of a Constituent System (CS)	28
20. RUI at the Cyber-Physical Layer	29
21. Connected RUS Interface Layers, Example Informational Layer	30
22. Example of Emergence in physical world	33
23. The Entity of a Two-Levels Hierarchy	34
24. Multi-level Hierarchy (Holarchy)	35
25. Holarchy relations	37
26. Informational Interactions	38
27. Types of Emergence	39
28. Example of dead lock	40
29. Relative Time for Three Concurrent Processes	45
30. Election algorithm -Bully	52
31. Failure Detectors represented in the table	55
32. Reduction algorithm	56
33. Algorithm S	57

34. <i>Algorithm Comparison – table</i>	66
35. <i>Hard Real Time versus Soft Real Time</i>	69
36. <i>State and Event</i>	71
37. <i>Temporal parameters are associated with real-time data</i>	72
38. <i>Tasks</i>	74
39. <i>State transition and ready queue</i>	75
40. <i>The System definition</i>	79
41. <i>The Model definition: relationships between model and system</i>	80
42. <i>The metamodel definition: relationships between metamodel and model</i>	81
43. <i>The Modeling Language definition</i>	83
44. <i>The classification of a modeling language and its companion viewpoints</i>	84
45. <i>SysML imported to Blockly</i>	88
46. <i>Aiding user to add new blocks through dropdown</i>	89
47. <i>Three ways to view a block</i>	90
48. <i>Viewpoints/building-blocks of a block can be enabled or disabled</i>	91
49. <i>Filtered view of SoS</i>	92
50. <i>Example of blocks related to requirements management</i>	93
51. <i>An example of a constraint where the member variable m_valid is checked</i>	94
52. <i>Detecting emergence in model through constraints</i>	95
53. <i>Model querying large models</i>	95
54. <i>Result of “return true;” query</i>	96
55. <i>Reusing an existing block (cs1) using links</i>	97
56. <i>Similar blocks can be grouped together</i>	97
57. <i>Example of behaviour for a service</i>	98
58. <i>Specifying the cardinality for CS – cs1</i>	98
59. <i>Sequence diagram in supporting facility tool using Blockly</i>	99
60. <i>Multi-Robot Lidar Sensor</i>	102
61. <i>System Architecture of Smart Farm</i>	103
62. <i>Two-Phase Commit protocol</i>	104
63. <i>Robot Visualizer</i>	105
64. <i>Lidar Sensor</i>	105
65. <i>Obstacle avoidance logic</i>	105
66. <i>Model Architecture of North & South Farm and Store house</i>	106