

Computação Orgânica*

Índice

1.	Introdução.....	2
2.	Definição.....	15
3.	Uma estrutura formal para computação orgânica	18
3.1.	Aspectos de implementação.....	23
4.	Exemplos de aplicação e formas de validação	24
5.	Referências bibliográficas	26

* Nota: Este material, a menos da Introdução (Seção 1), está predominantemente baseado em BRANKE *et al.* (2007).

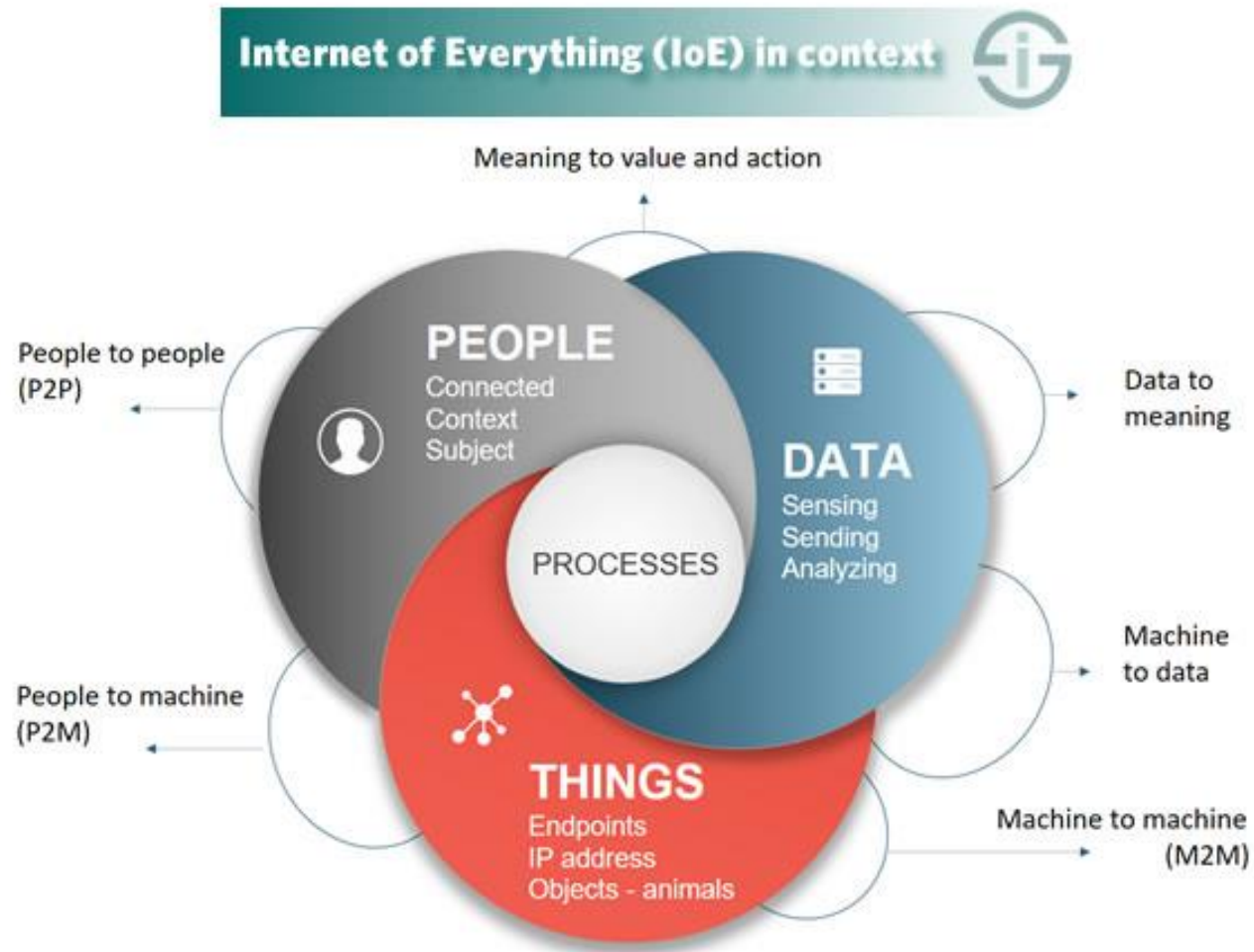
1. Introdução

- Amplia-se a cada dia o número de dispositivos computacionais flexíveis, multifuncionais e interligados em rede.
- Adicione-se a isso o incremento exponencial na capacidade de processamento e memória de cada dispositivo individual, sustentado pela lei de Moore.
- Desse modo, espera-se que as novas gerações de sistemas computacionais ganhem complexidade, **evoluam de forma auto-organizada** e imponham novos desafios aos seus projetistas e também aos seus usuários.
- Caminha-se, de forma irreversível, para a **Tecnologia do Relacionamento**, onde o conhecimento deriva de múltiplas fontes e requer integração, os objetivos são múltiplos e conflitantes e as interfaces são customizadas.
- Dispositivos computacionais equipados com recursos de comunicação sem fio vão interagir e trocar informação numa escala nunca antes imaginada.

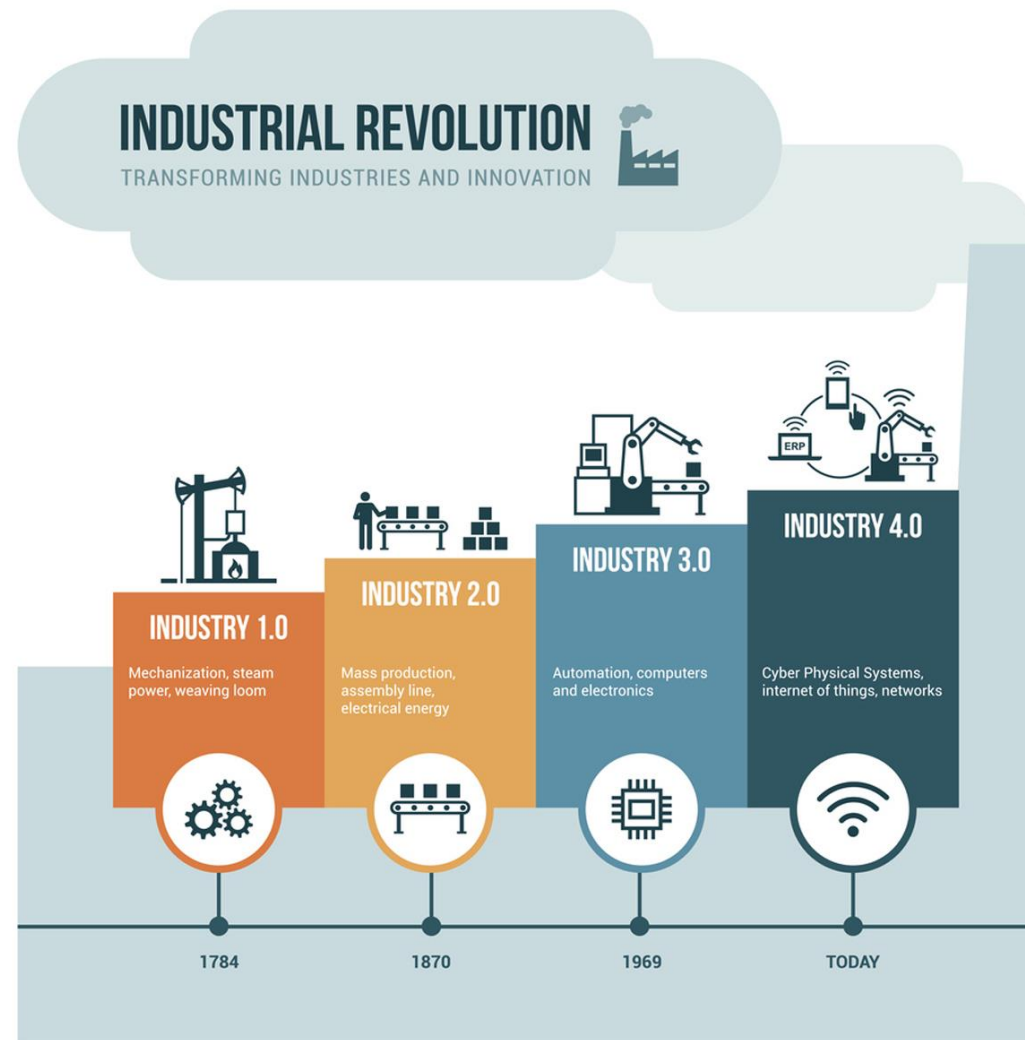
- É a chamada *Internet of Things* (IoT) (GUBBI *et al.*, 2013; VERMESAN & FRIESS, 2013), que já pode ser estendida para a *Internet of Everything* (IoE).



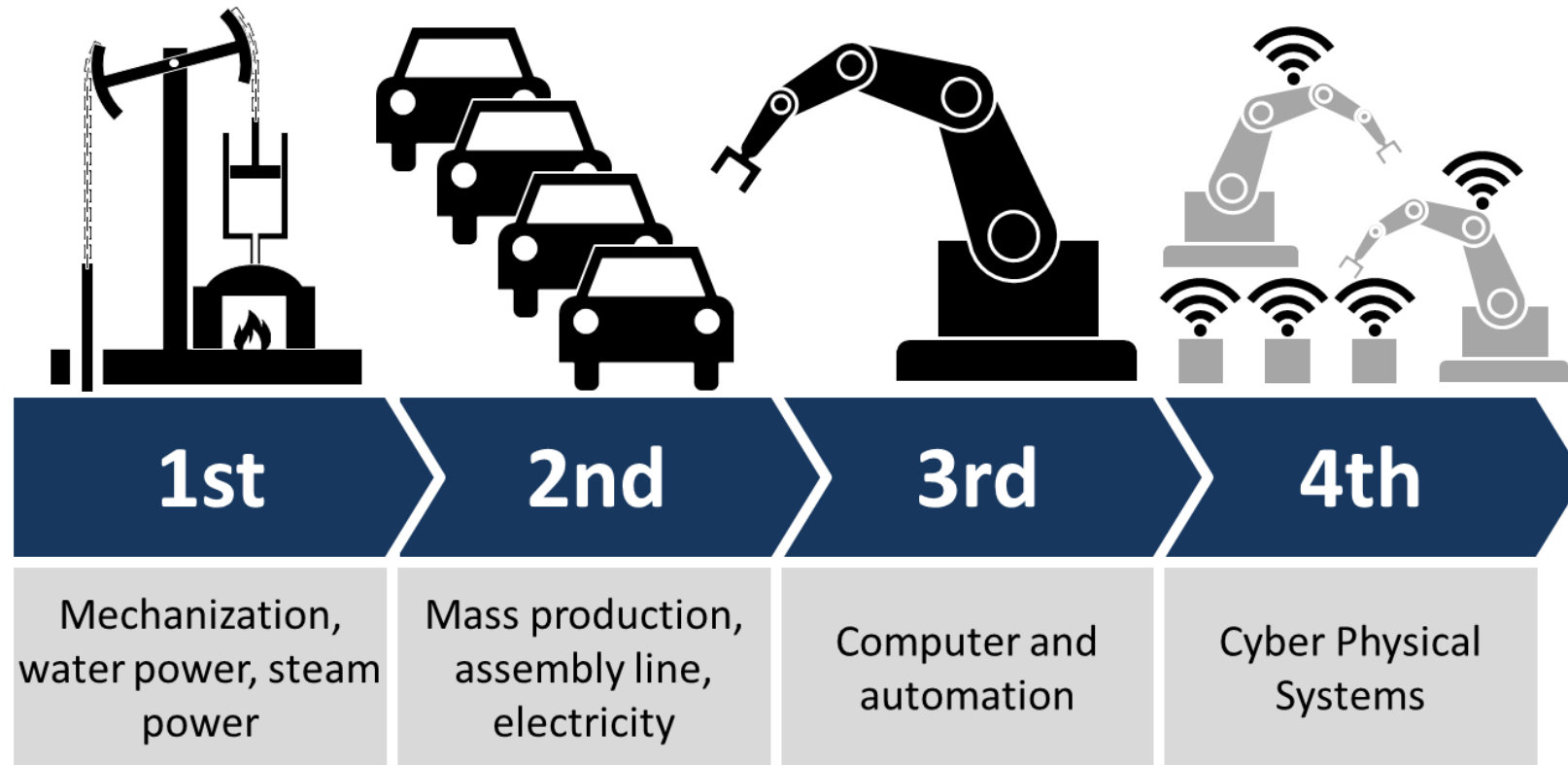
Fonte: SINTEF – A non-commercial organization



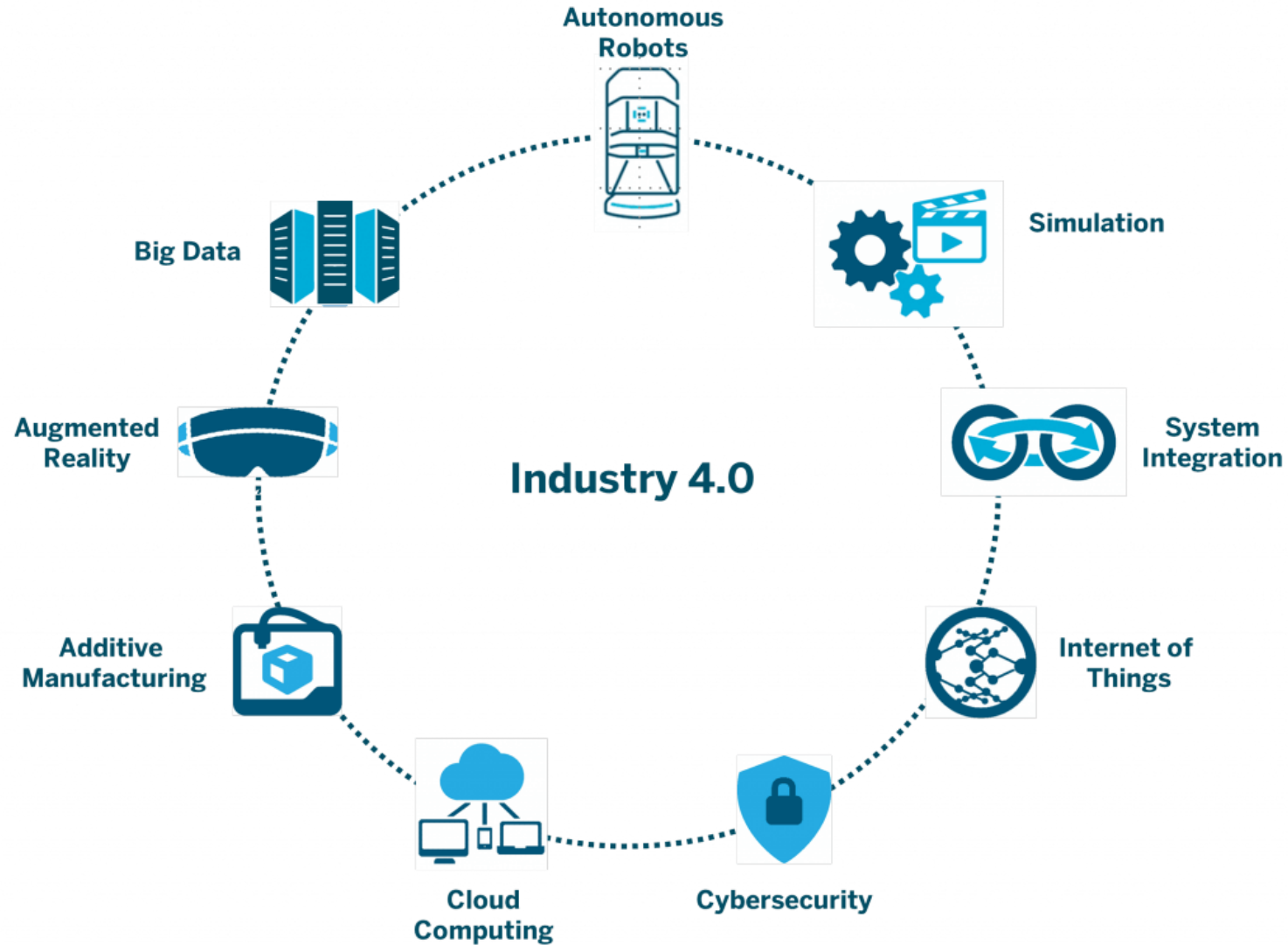
Fonte: <https://www.i-scoop.eu/internet-of-things-guide/smart-internet-things-facts-benefits/>



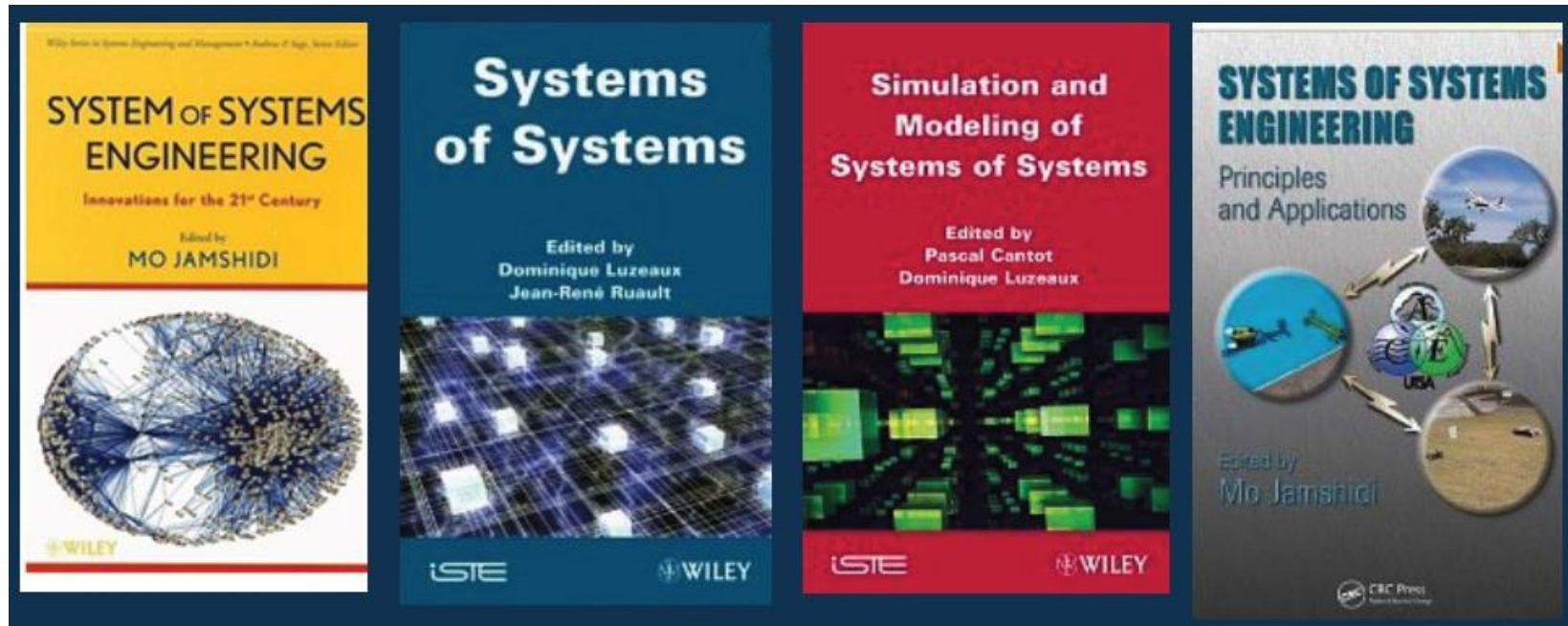
Fonte: <http://www.aberdeenessentials.com/opspro-essentials/industry-4-0-industrial-iot-manufacturing-sneak-peek/>



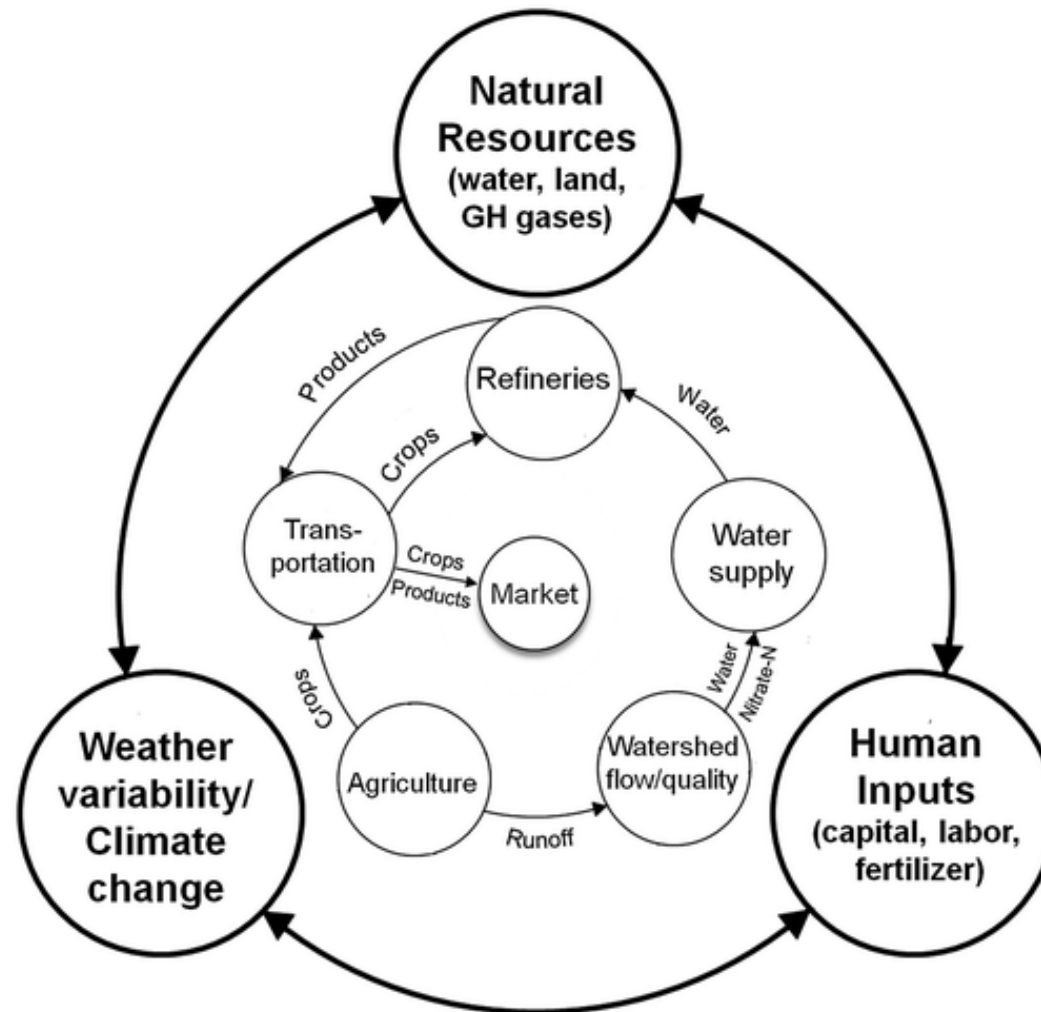
Fonte: https://en.wikipedia.org/wiki/Industry_4.0



Fonte: <http://www.aethon.com/industry-4-0-means-manufacturers/>



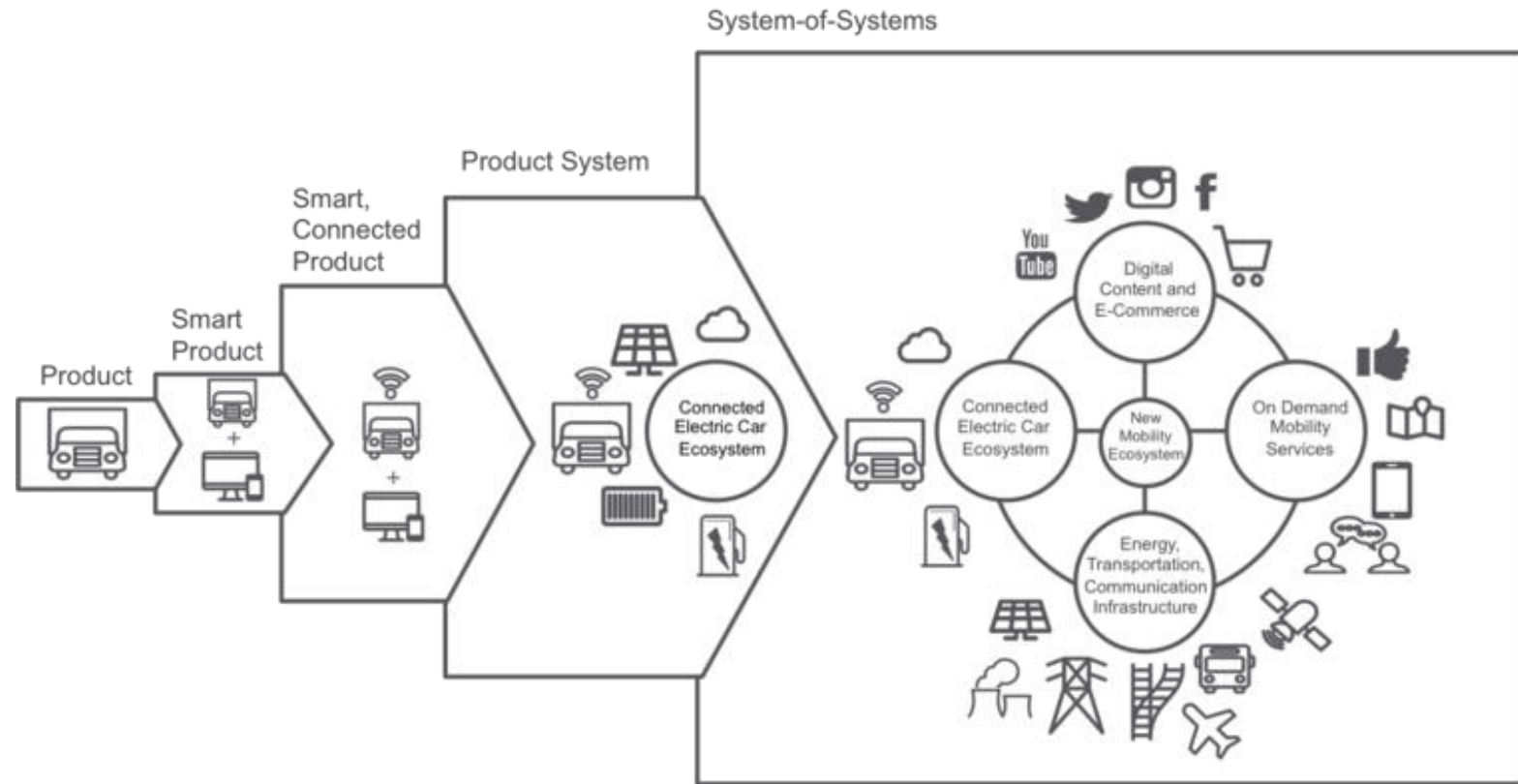
- *System of systems is a collection of task-oriented or dedicated systems that pool their resources and capabilities together to create a new, more complex system which offers more functionality and performance than simply the sum of the constituent systems. The methodology for defining, abstracting, modeling, and analyzing system of systems problems is typically referred to as **system of systems engineering**. (Fonte: https://en.wikipedia.org/wiki/System_of_systems)*



Modelo de sistema de sistemas para a produção de biocombustíveis

Fonte: <http://ascelibrary.org/doi/abs/10.1061/%28ASCE%29IS.1943-555X.0000238>

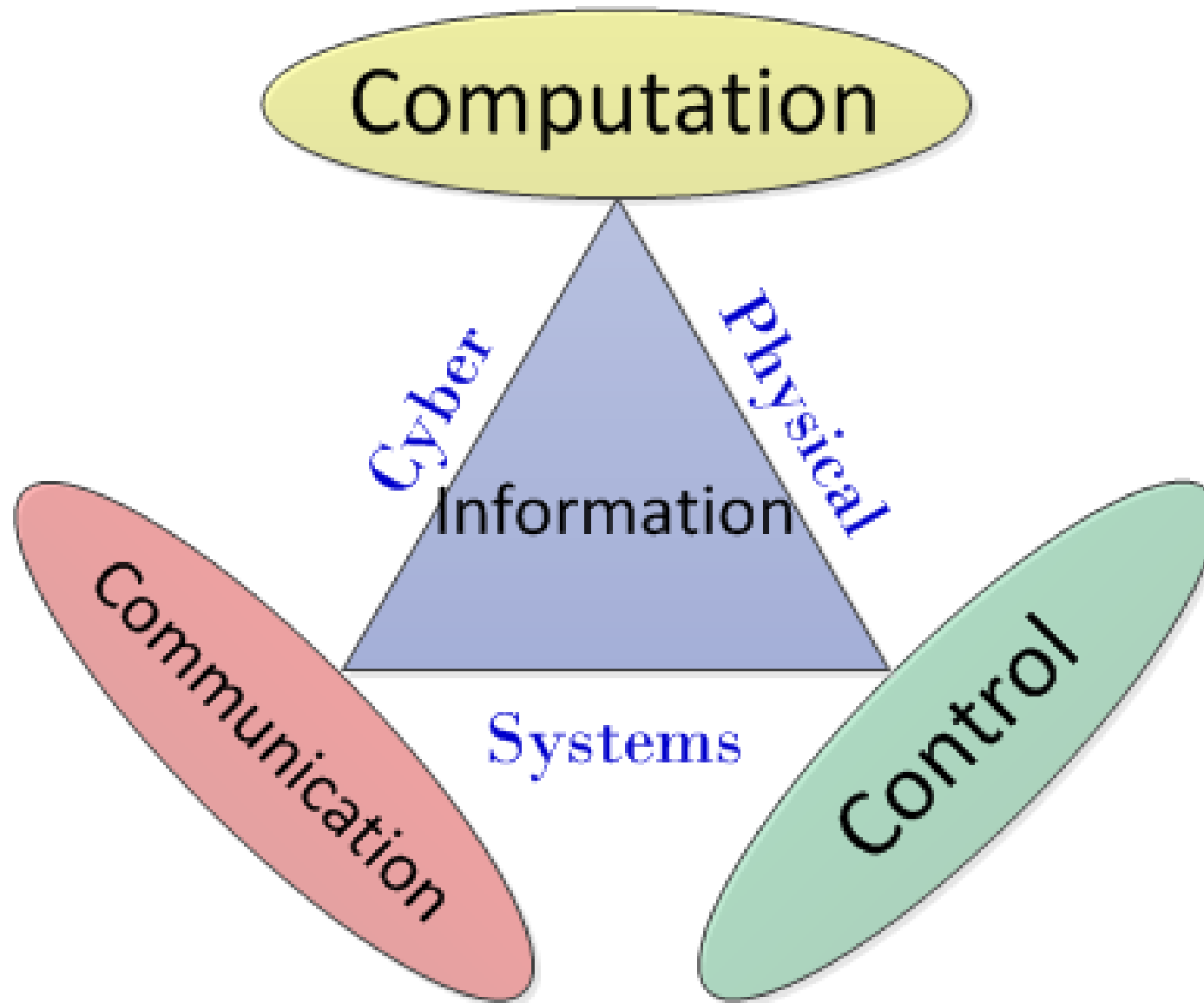
- Um outro exemplo preliminar desse cenário de evolução tecnológica pode ser tomado a partir dos sistemas já embutidos em um automóvel: (1) controle de combustão e emissão de poluentes, (2) monitoramento de variáveis e de pontos de operação, (3) sistemas de segurança (como *air bags*), (4) condicionamento de ar, (5) sistema de navegação, (6) sistema de auxílio para estacionar o veículo, (7) controle de cruzeiro, (8) sistema de entretenimento aos passageiros.
- Esses sistemas de um automóvel estão interligados em uma complexa rede de comunicação interna e o seu desenvolvimento, com a incorporação de novas funcionalidades, é continuado. Num futuro próximo, os automóveis tendem a se adaptar a diferentes motoristas e condições das pistas, estabelecer ampla troca de informações com outros automóveis e com centros de monitoramento, além de integrar dispositivos em redes formadas pelos próprios automóveis.
- Esta expansão tecnológica, se por um lado traz facilidades aos usuários, por outro lado representa um grande desafio aos projetistas desses sistemas.



De um produto a um sistema de sistemas

Fonte: <https://www.newmobilityconsulting.com/>

- Como ser capaz de projetar tais sistemas amplamente distribuídos, complexos e densamente conectados, e como torná-los confiáveis e de fácil uso?
- É evidente que o projetista não será capaz de prever todas as possíveis configurações e nem mesmo indicar o comportamento mais adequado em todos os casos e em detalhes.
- O usuário também deve ser liberado da função de ajuste de parâmetros de baixo nível, de modo que a sua influência na operação desses sistemas se dê com base na definição de metas de alto nível.
- Essencialmente, a complexidade crescente dos sistemas computacionais requer novos princípios de projeto, pois eles tendem a exibir propriedades emergentes, que não podem ser devidamente antecipadas.
- A computação orgânica, a qual será devidamente formalizada na próxima seção, é apenas uma das formas de conceber e modelar *cyber physical systems* e *system of systems*.



May 2016 | Volume 104 | Number 5

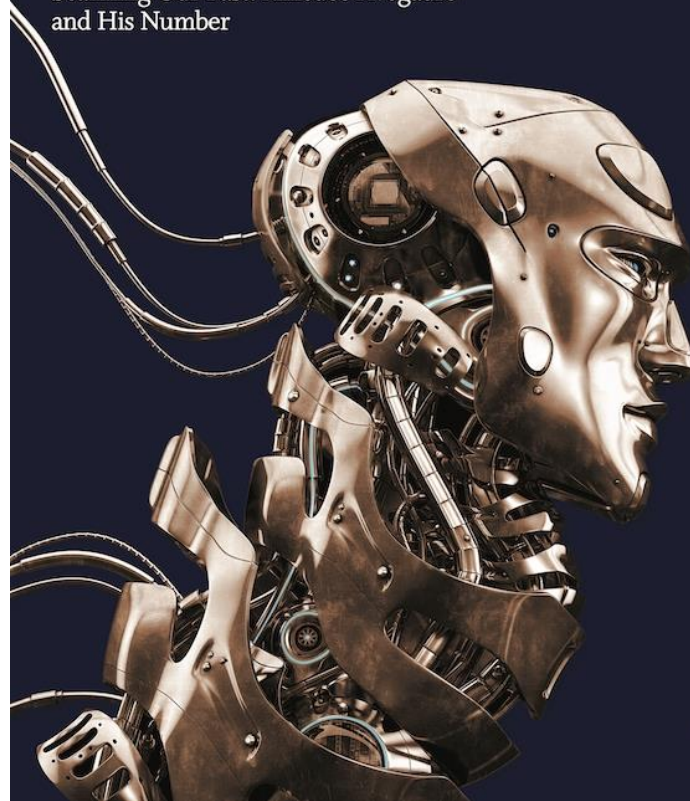
Proceedings OF THE IEEE

SPECIAL ISSUE

Industrial Cyber-Physical Systems

Point of View: Technology Versus Privacy Issues
in Preventing Distracted Driver Accidents

Scanning Our Past: Amedeo Avogadro
and His Number



 IEEE

2. Definição

- A computação orgânica tem por objetivo lidar com os desafios impostos por sistemas complexos distribuídos, **interpretando-os como se fossem organismos vivos**. São atribuídas a esses sistemas as chamadas auto-x-propriedades: auto-organização, auto-configuração, auto-reparo, auto-entendimento, auto-proteção e auto-adaptação.
- Busca-se, assim, simplificar a tarefa de projeto do sistema, visto que não se faz mais necessário especificar o comportamento de baixo nível do sistema para todos os cenários possíveis de operação. Também deixa de ser tarefa do projetista/usuário a manipulação de múltiplos parâmetros que influenciam de forma intrincada o desempenho do sistema.
- Basta ao projetista/usuário especificar **um número reduzido de objetivos de alto nível**.

- Com isso, o sistema ganha um certo grau de autonomia, devendo reagir de forma apropriada a situações novas e, possivelmente, imprevisíveis.
- Apesar de sua complexidade, criaturas vivas apresentam robustez a alterações nas condições internas e externas, exibindo habilidades de aprendizado e adaptação a ambientes incertos e dinâmicos.
- Logo, **abordar a complexidade de projeto, evolução e operação de sistemas computacionais como se eles fossem criaturas vivas conduz a uma proposta alternativa àquela que requer a sua especificação completa e a priori.**
- Isso requer que tais sistemas possuam graus adicionais de liberdade, mas, por outro lado, traz algumas desvantagens: (1) para aprender, o sistema precisa cometer erros, o que pode tornar sua reação mais lenta a novos cenários; (2) os comportamentos e as próprias reações são menos previsíveis; (3) como os sistemas são auto-organizáveis, é mais difícil controlá-los.

- Conclusão: Deve haver um balanço entre os processos *bottom-up* de “criatividade auto-organizada” e o controle *top-down*.
- Na computação orgânica, os sistemas computacionais são capazes de exibir aprendizado on-line, sobreviver a ataques e a falhas internas de componentes, se adaptar aos usuários e reagir de forma apropriada mesmo a situações imprevisíveis.
- Os objetivos da computação orgânica (CO) são muito similares àqueles associados à computação autônoma (CA) (do inglês *autonomic computing*) (KEPHART & CHESSE, 2003; STERRITT, 2005).
- A principal diferença é que a CA está mais voltada para aplicações em arquiteturas de servidores, enquanto que a CO engloba todo tipo de sistema computacional distribuído e auto-organizado (GÜDEMANN *et al.*, 2008; SEEBACH *et al.*, 2007).
- Embora esteja ainda se estabelecendo como uma área de pesquisa, a CO é um campo de estudo interdisciplinar que busca gerenciar o crescimento continuado de complexidade dos sistemas tecnológicos.

3. Uma estrutura formal para computação orgânica

- A estrutura mais empregada para o projeto de sistemas de computação orgânica é a chamada arquitetura observador-controlador (MÜLLER-SCHLOER *et al.*, 2004), conforme apresentado nas Figuras 1 e 2. Ela é similar ao ciclo MAPE (monitorar-analisar-planejar-executar) adotado para a computação autonômica, e pode ser interpretado como **um processo de auto-organização controlada** (PROKOPENKO, 2009).
- O observador monitora todo o sistema e é capaz de detectar tendências. Se um problema (Exemplo: erro, operação anormal) é detectado ou antecipado, o controlador age no sentido de resolver o problema ou minimizar os seus efeitos sobre todo o sistema, levando em conta a natureza emergente, distribuída e auto-organizada do comportamento complexo e global do sistema (MÜLLER-SCHLOER *et al.*, 2011).

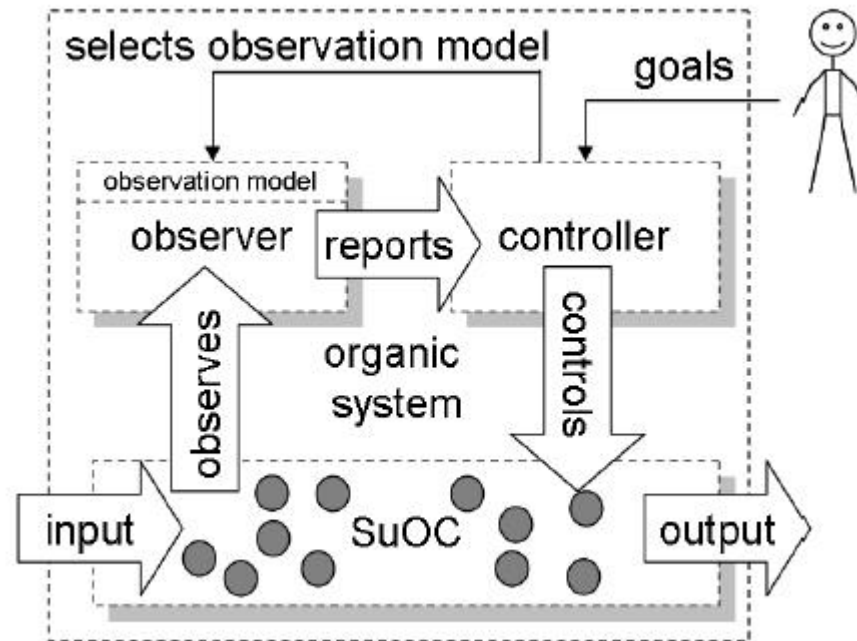


Figura 1 – Arquitetura observador-controlador em computação orgânica (SuOC: *System under Observation and Control*)

- Uma perspectiva mais detalhada é apresentada na Figura 2, sendo que as Figuras 1 e 2 foram extraídas de BRANKE *et al.* (2007).

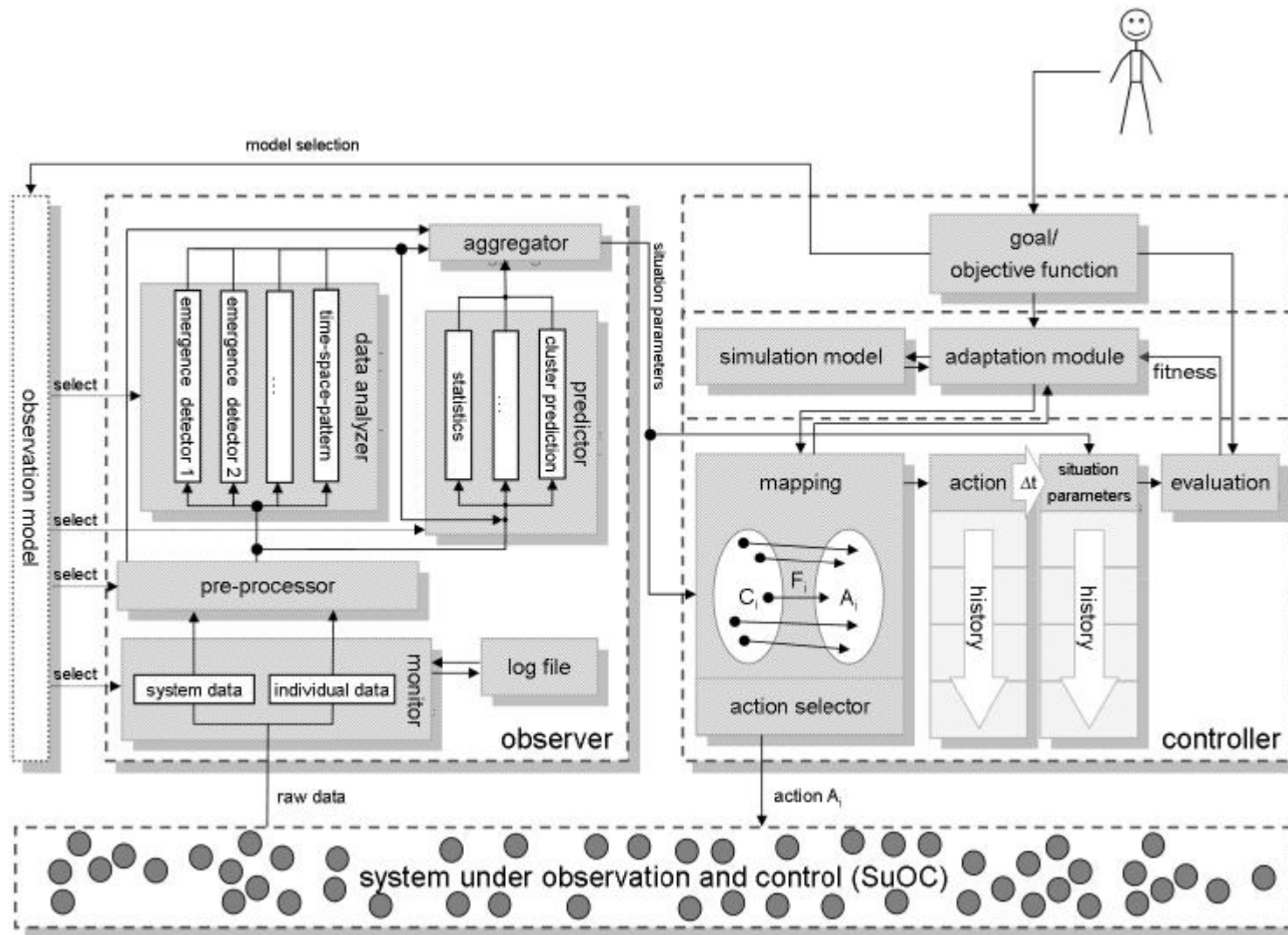


Figura 2 – Uma visão mais detalhada da arquitetura observador-controlador

- Cabe mencionar que o sistema sob observação e controle (SuOC) continua a operar mesmo se observador e controlador forem desativados.
- É esperado que o controlador atue apenas quando necessário e seu poder de atuação se restringe a um subconjunto de parâmetros do SuOC.
- O processo de observação consiste dos seguintes passos: monitoramento, pré-processamento, análise de dados, predição e agregação.
- Um modelo de observação (*observation model*) caracteriza o observador ao selecionar atributos do monitor, do pré-processador, do analisador de dados e do preditor.
- O preditor recebe como entrada os dados provenientes do pré-processador e do analisador de dados e os utiliza para predizer o estado futuro do sistema.
- As saídas do preditor, do pré-processador e do analisador de dados são agregadas e formam a entrada para o controlador, denominada de situação (*situation parameters*).

- Dentre os principais módulos do controlador está o selecionador de ações (*action selector*), o qual seleciona a ação mais adequada para a situação corrente. Sua reação deve ocorrer em tempo real, o que implica que o processamento interno deve ser suficientemente rápido. **Algumas implementações empregam sistemas classificadores com aprendizado** (*learning classifier systems*) aqui.
- O módulo de adaptação implementa capacidades de aprendizado e de planejamento, com o propósito de maximizar os objetivos formulados pelo projetista ou pelo usuário corrente. A avaliação de ações implementadas permite o aprendizado de melhorias no mapeamento situação → ação.
- A avaliação é executada tomando uma base de dados históricos, o que permite atribuir certas mudanças de estado do sistema a ações específicas.
- Conectado ao módulo de adaptação, pode existir um modelo de simulação, permitindo inferir as consequências de certas ações antes delas serem efetivamente implementadas no sistema real.

3.1. Aspectos de implementação

- A arquitetura apresentada pode ser customizada junto a diferentes cenários de aplicação, produzindo as seguintes versões:
 - Centralizada: um observador-controlador para todo o sistema;
 - Descentralizada: um observador-controlador para cada elemento do sistema;
 - Multinível: um observador-controlador para cada elemento do sistema e também para todo o sistema.
- Na estrutura multinível, podem surgir holarquias (do inglês *holarchy*), em que um elemento é ao mesmo tempo o todo e uma parte do todo.
- Evoluções tecnológicas, como a perspectiva *Network-on-a-chip* (NoC) para a comunicação de múltiplos cores de processamento, criam a possibilidade de um hardware que irá requerer o paradigma da computação orgânica para viabilizar a sua operação.

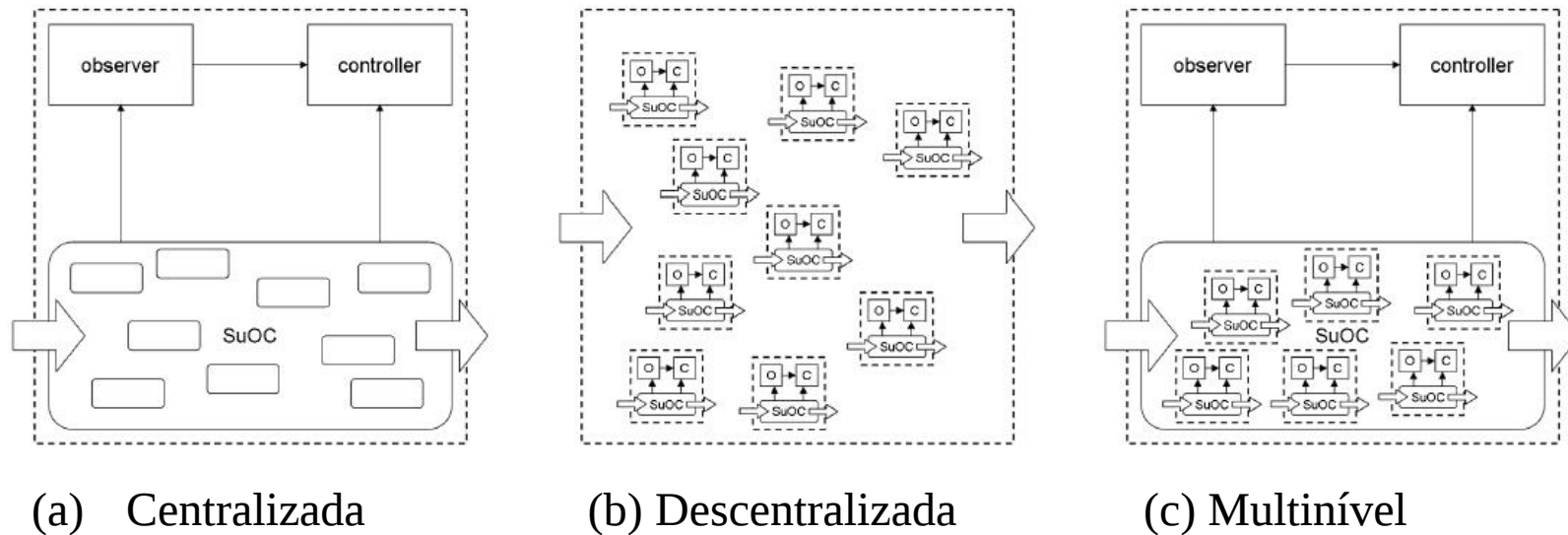


Figura 3 – Possibilidades de implementação da arquitetura observador-controlador

4. Exemplos de aplicação e formas de validação

- As aplicações já realizadas vão de *smart grids* ao controle auto-organizado de tráfego, passando pela coordenação auto-organizada de múltiplos robôs, os quais podem ser definidos como *cyber physical systems* e/ou *system of systems*.

- A computação orgânica pode ser considerada uma **nova estratégia de projeto de engenharia**, pois o sistema projetado deve ser capaz de auto-organizar diversos aspectos de seu comportamento, fazendo isso de forma autônoma e, possivelmente, na presença de interferência externa.
- Essencialmente, o sistema deve manter um bom nível de desempenho mesmo na presença de ambientes variantes no tempo, mudanças de objetivos, de funcionalidades requeridas e de relações entre os sistemas envolvidos. Para tanto, ele deve exibir (parte de) as auto-x-propriedades.
- É evidente que **a validade da computação orgânica para a solução de problemas complexos será comprovada caso seja possível demonstrar formalmente ou ao menos fornecer evidências empíricas de que a solução fornecida por um sistema orgânico é mais confiável, mais robusta e mais eficaz do que aquela fornecida por um sistema não-orgânico.**
- Questões de escalabilidade e tendências tecnológicas devem ser consideradas.

5. Referências bibliográficas

- AGARWAL, A. & HARROD, B. (2006), “Organic Computing”, MIT and DARPA Technical Report.
- BRANKE, J., MNIF, M., MÜLLER-SCHLOER, C., PROTHMANN, H., RICHTER, U., ROCHNER, F. & SCHMECK, H. (2007), “Organic Computing – Addressing Complexity by Controlled Self-organization”, Proceedings of the Second International Symposium on Leveraging Applications of Formal Methods, Verification and Validation, pp. 185-191.
- DOURSAT, R., SAYAMA, H. & MICHEL, O. (2012) “Morphogenetic Engineering – Toward Programmable Complex Systems”, Understanding Complex Systems Series, Springer.
- GUBBI, J., BUYYA, R., MARUSIC, S. & PALANISWAMI, M. (2013) “Internet of Things (IoT): A vision, architectural elements, and future directions”, Future Generation Computer Systems, vol. 29, pp. 1645-1660.
- GÜDEMANN, M., NAFZ, F., ORTMEIER, F., SEEBACH, H. & REIF, W. (2008) “A Specification and Construction Paradigm for Organic Computing Systems”, Proceedings of the Second IEEE International Conference on Self-Adaptive and Self-Organizing Systems, pp. 233-242.
- KEPHART, J.O. & CHESS, D.M. (2003), “The vision of autonomic computing”, IEEE Computer, vol. 36, no. 1, pp. 41-50.

- MÜLLER-SCHLOER, C., SCHMECK, H. & Ungerer, T. (Eds.) (2011) “Organic Computing – A Paradigm Shift for Complex Systems”, Springer.
- MÜLLER-SCHLOER, C., VON DER MALSBURG, C. & WÜRTZ, R.P. (2004), “Organic Computing”, Informatik Spektrum, vol. 27, no. 4, pp. 332-336.
- PROKOPENKO, M. (2009), “Guided self-organization”, HFSP Journal, vol. 3, no. 5, pp. 287-289.
- SCHMECK, H. (2005), “Organic Computing – A New Vision for Distributed Embedded Systems”, Proceedings of the Eighth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC’2005), pp. 201-203.
- SCHMECK, H., MÜLLER-SCHLOER, C., ÇAKAR, E., MNIF, M. & RICHTER, U. (2010), “Adaptivity and Self-Organization in Organic Computing Systems”, ACM Transactions on Autonomous and Adaptive Systems, vol. 5, no. 3, Article 10.
- SEEBACH, H., ORTMEIER, F. & REIF, W. (2007) “Design and Construction of Organic Computing Systems”, Proceedings of the IEEE Congress on Evolutionary Computation (CEC’2007), pp. 4215-4221.
- STERRITT, R. (2005), “Autonomic Computing”, Innovations in systems and software engineering, vol. 1, no. 1, pp. 79-88.
- VERMESAN, O. & FRIESS, P. (Eds.) (2013) “Internet of Things – Converging Technologies for Smart Environments and Integrated Ecosystems”, River Publishers.
- WÜRTZ, R.P. (Ed.) (2008) “Organic Computing”, Understanding Complex Systems Series, Springer.