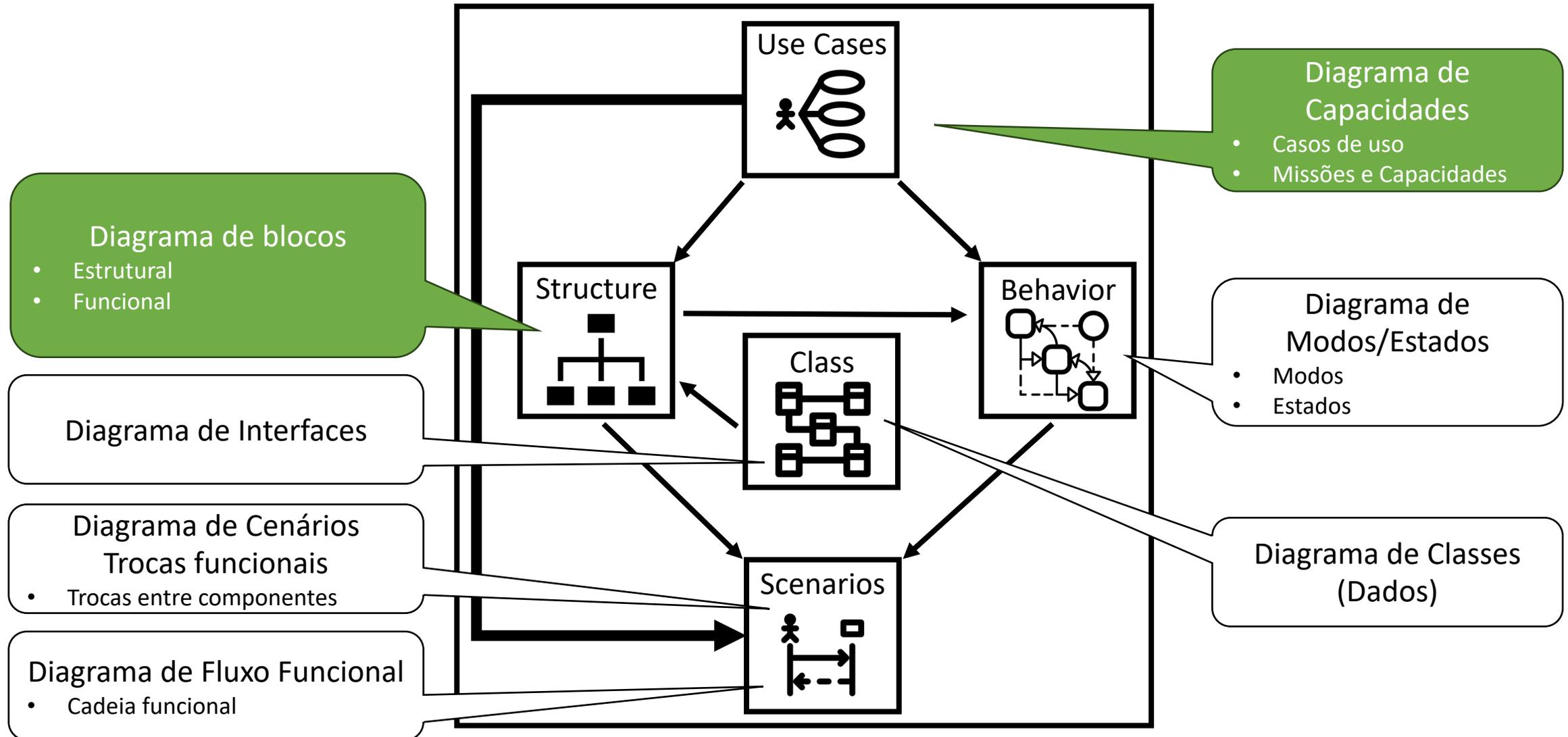




ROTEIRO DOS DIAGRAMAS





	SEMANA	TEORIA	INDIVIDUAL	PESO	GRUPO	PESO
Teoria, Linguagem e Lógica 	1	Estrutura do Curso e Apresentações Pessoais				
	04-Mar	O que é Engenharia de Sistemas? INCOSE	AI-01 - Leitura/Resumo Cap 1 - HB INCOSE	5%	AG-01 - Entrega dos grupos.	0%
	08-Mar	Contexto: Aeronave x Sistemas do Dom. Aéreo				
	2	Gramática: Representações clássicas				
	11-Mar		AI-02 - Leitura/Resumo paper sobre representações clássicas.	5%	AG-02 -	0%
	15-Mar	Lógica: Diagrama de Blocos				
	3	Gramática: Arquitetura e Visões				
	18-Mar	Gramática: Funções	AI-03 - Exercícios sobre arquitetura funcional e diagrama de blocos.	5%	AG-03 - Resumo sobre arquitetura funcional.	10%
	22-Mar	Lógica: Diagrama de Classes				
	4*	Gramática: Stakeholders				
	25-Mar	Gramática: Ciclo de Vida e CONOPS	AI-04 - Exercícios sobre diagrama de casos de uso	5%	AG-04 - Resumo sobre stk, ciclo de vida e CONOPS	10%
	29-Mar	Lógica: Diagrama de Casos de Uso				
	5	Gramática: Requisitos				
	01-Apr	Gramática: Verificação e Validação	AI-05 - Exercício de escrita de requisitos.	10%	AG-05 - Resumo sobre requisitos vindos da parte de segurança (safety) do RBAC	10%
	05-Apr	Contexto: ANAC (RBACs) 				
	6	Lógica: Diagrama de Estados				
	08-Apr		AI-06 - Exercício sobre diagrama de estados.	10%	AG-06 - Revisita do CONOPs do DECEA usando storyboards	15%
	12-Apr	Contexto: DECEA (ICAs/DCAs) 				
	7	Lógica: Diagrama de Sequência				
	15-Apr		AI-07 - Exercício sobre diagrama de sequencia.	0%	AG-07 - Representação do CONOPs usando Diagrama de Sequencia.	20%
19-Apr	Temas do projeto do segundo semestre e P1					
8	P1					
22-Apr		AI-08(P1) - Teoria e linguagem (Pres/Consulta - sem chatGPT)	60%	AG-01(P1) - Apresentação Gravada: 20min	40%	
26-Apr						
				100%		105%



IEA-P – DEPARTAMENTO DE PROJETOS
(PROJECT DEPARTMENT)

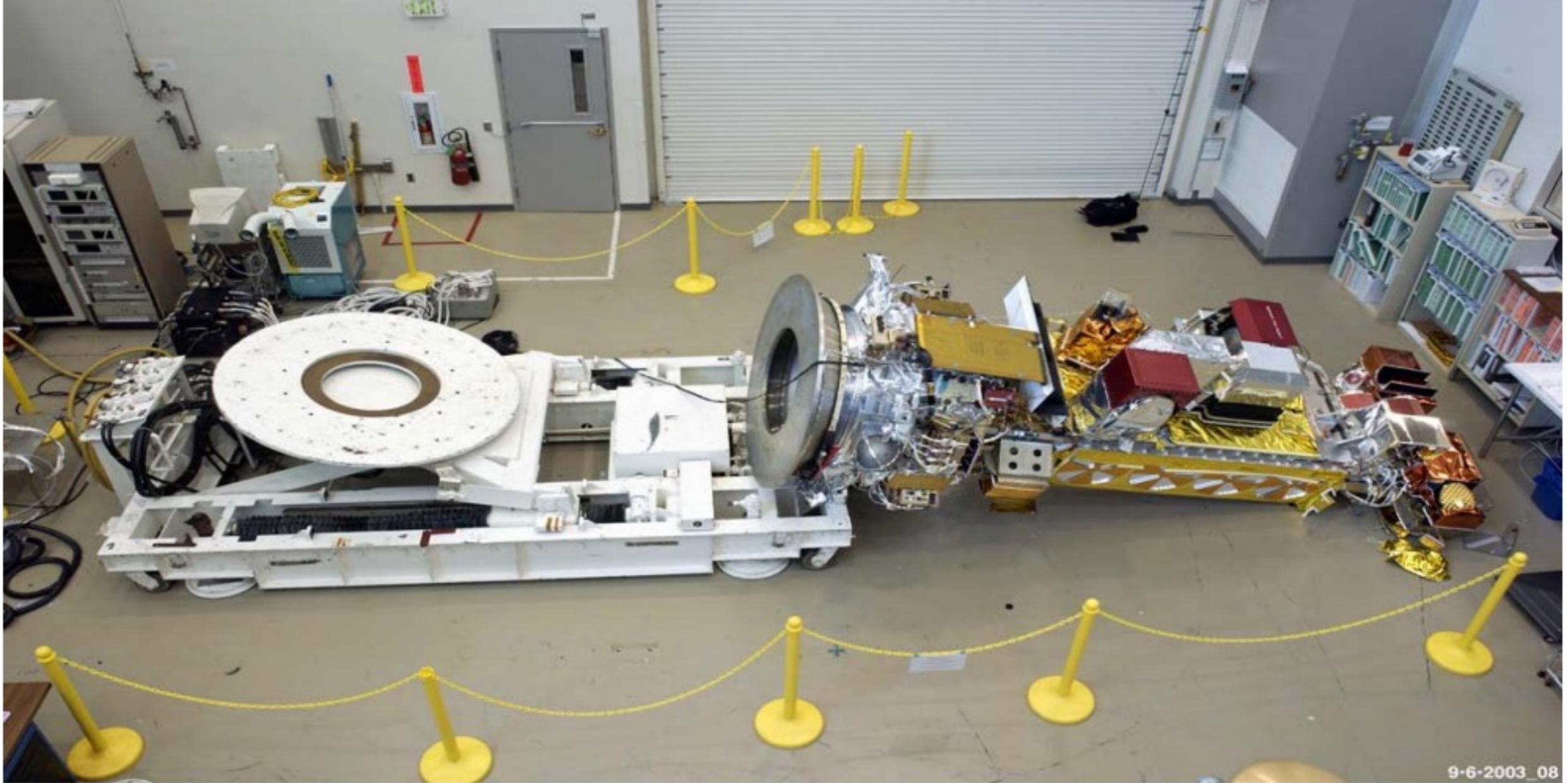
Verificação & Validação e Diagrama de Estados

[2024]

Prof. Dr. Christopher S. Cerqueira



NEM TUDO SÃO FLORES



https://www.nasa.gov/pdf/65776main_noaa_np_mishap.pdf

FIGURE 3-1: NOAA N-PRIME FELL OFF THE TURNOVER CART





<https://www.theverge.com/2019/5/2/18518176/boeing-737-max-crash-problems-human-error-mcas-faa>





PROCESSO DE VERIFICAÇÃO GLOBAL



OBJETIVOS

- O objetivo geral do processo V&V é demonstrar, por meio de um processo dedicado, que o produto entregável atende aos requisitos especificados e conceito de operação descrito.
- **Os objetivos do processo são os seguintes:**
 - a. **PROJETO:** demonstrar a **qualificação do projeto e do desempenho**, como atendendo aos requisitos especificados nos níveis especificados;
 - b. **FABRICAÇÃO:** para garantir que o **produto está de acordo com o projeto**, está livre de defeitos de fabricação e aceitável para uso;
 - c. **ASSEMBLÉIA:** para **confirmar a integridade e o desempenho** do produto em etapas específicas do ciclo de vida do projeto.
 - d. **INTEGRAÇÃO:** Confirmar que o sistema (incluindo ferramentas, procedimentos e recursos) **pode cumprir os requisitos** da missão.



ABORDAGEM

- A abordagem básica deve ser desenvolvida através de um processo de iteração, com base em **considerações técnicas / de custo / cronograma**, que define **o que, como, onde e quando**:
 1. **Identificar** um conjunto consistente de requisitos de projeto verificáveis que podem ser submetidos ao processo de verificação;
 2. **Selecionar** os métodos de verificação;
 3. **Selecionar** os níveis de verificação e a filosofia do modelo associado;
 4. **Selecionar** as instalações;
 5. **Identificar** os recursos necessários;
 6. **Identificando** as etapas e os eventos em que as ações de VV serão implementadas.



VERIFICAÇÃO



VERIFICAÇÃO

- **A VERIFICAÇÃO** de um produto mostra **prova de conformidade com os requisitos** - o produto pode atender a cada declaração "deve" como comprovada através do desempenho de um teste, análise, inspeção ou demonstração (ou combinação destes).
- **VERIFICAÇÃO é um processo de confirmação de que um requisito ou sistema está em conformidade.**
 - Em outras palavras, a verificação do sistema faz a pergunta: **o sistema atende aos seus requisitos?**
 - A verificação de requisitos faz a pergunta: **o sistema realmente atende a esse requisito específico?**



VERIFICAÇÃO

- A **VERIFICAÇÃO** refere-se aos **requisitos aprovados e pode ser realizada em diferentes fases do ciclo de vida do produto.**
 - As especificações, desenhos, listas de peças e outras documentações de configuração aprovadas estabelecem a linha de base (baseline) de configuração desse produto (que pode ser modificada posteriormente).
 - Sem um baseline e controles de configuração apropriados, **modificações posteriores podem ser dispendiosas ou causar grandes problemas de desempenho.**



VALIDAÇÃO



VALIDAÇÃO

- A **VALIDAÇÃO** de um produto mostra que o produto **cumpre o propósito pretendido no ambiente pretendido** - que atende às expectativas do cliente e de outras partes interessadas, conforme demonstrado pelo desempenho de um teste, análise, inspeção ou demonstração.
- **VALIDAÇÃO** é um processo de confirmação de que um conjunto de requisitos, design ou sistema **atende à intenção** do desenvolvedor ou cliente.



VALIDAÇÃO

- VALIDAÇÃO refere-se ao documento **ConOps**.
- A **VALIDAÇÃO** é conduzida em **condições realistas (ou condições simuladas)** em **produtos finais** com o objetivo de determinar a eficácia e adequação do produto para **uso em operações de missão por usuários típicos**.
- A **VALIDAÇÃO** pode ser realizada em cada fase de desenvolvimento usando produtos de fase (por exemplo, modelos) e não apenas na entrega usando produtos finais.



VALIDAÇÃO X VERIFICAÇÃO

- **VERIFICANDO UM SISTEMA:**

Construindo o sistema corretamente: garantindo que o sistema esteja em conformidade com os requisitos do sistema.



- **VALIDANDO UM SISTEMA:**

Construindo o sistema certo: certificando-se de que o sistema faz o que deve fazer em seu ambiente operacional.



MÉTODOS DE COMPROVAÇÃO



INSPEÇÃO

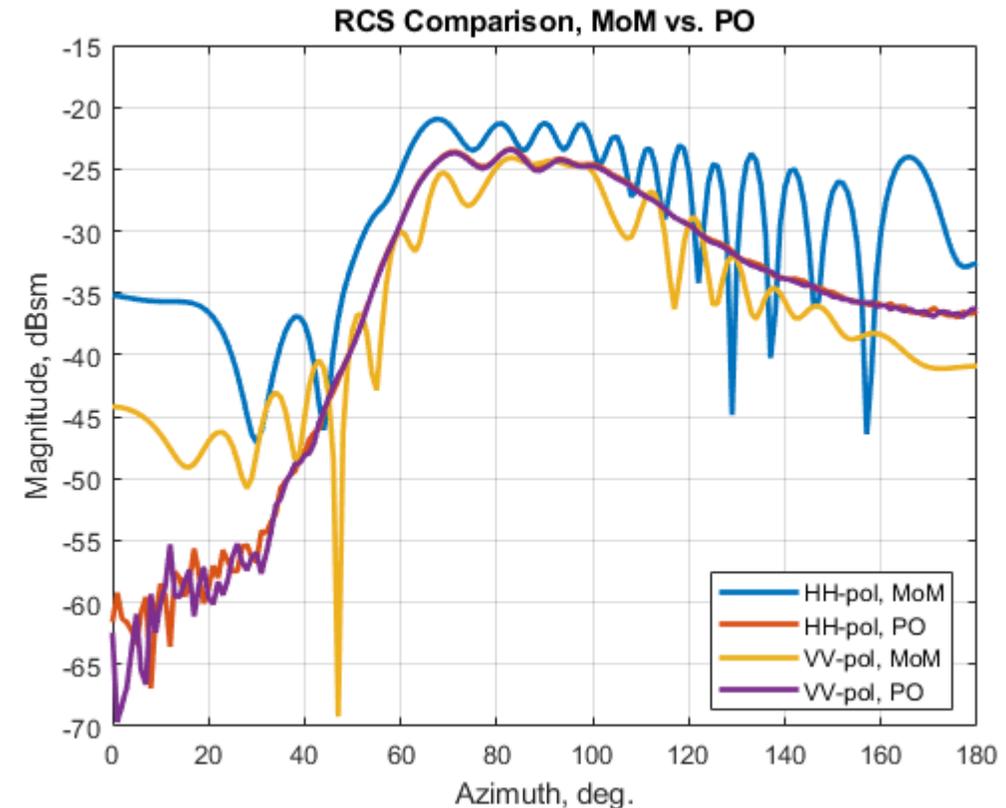
- **Exame visual de desenhos ou dados**
- **Examinando um atributo físico direto do próprio item**
 - Dimensões
 - Peso
 - Características físicas
 - Cor ou marcações





ANÁLISE

- **Avaliação dos dados por técnicas analíticas:**
 - análise de engenharia de sistemas
 - estatística
 - análise qualitativa
 - modelagem análoga
 - similaridade
 - simulação
- Também pode ser avaliação de dados derivados de verificações de nível inferior.





SIMILARIDADE

- Avaliar, por revisão de dados de aceitação prévia, que um Sistema, ou sua fabricação, é semelhante ou idêntico, a outro sistema que tenha sido previamente qualificado.
- Geralmente considerado um subconjunto da Análise
- Não é usado quando uma ou ambas as seguintes condições existirem:
 - O mesmo item utilizado na avaliação foi verificado por meio da semelhança como método.
 - Itens de criticidade (ou seja, perda de vida ou ferimentos graves)





DEMONSTRAÇÃO

- Mostrando que o uso ou operação alcança os resultados sem coleta de dados detalhada. Utilize modelos físicos ou maquetes.
- Muitas vezes associadas às "ilidades" (acessibilidade, transportabilidade, facilidade de manutenção, etc.)





VALIDAÇÃO DE REGISTROS

- O uso de registros de fabricação fornecidos pelo fornecedor para garantir que os requisitos tenham sido atendido.
- Frequentemente usado para produtos COTS ou produtos comprados de acordo com padrões.





ESCOLHENDO O MÉTODO

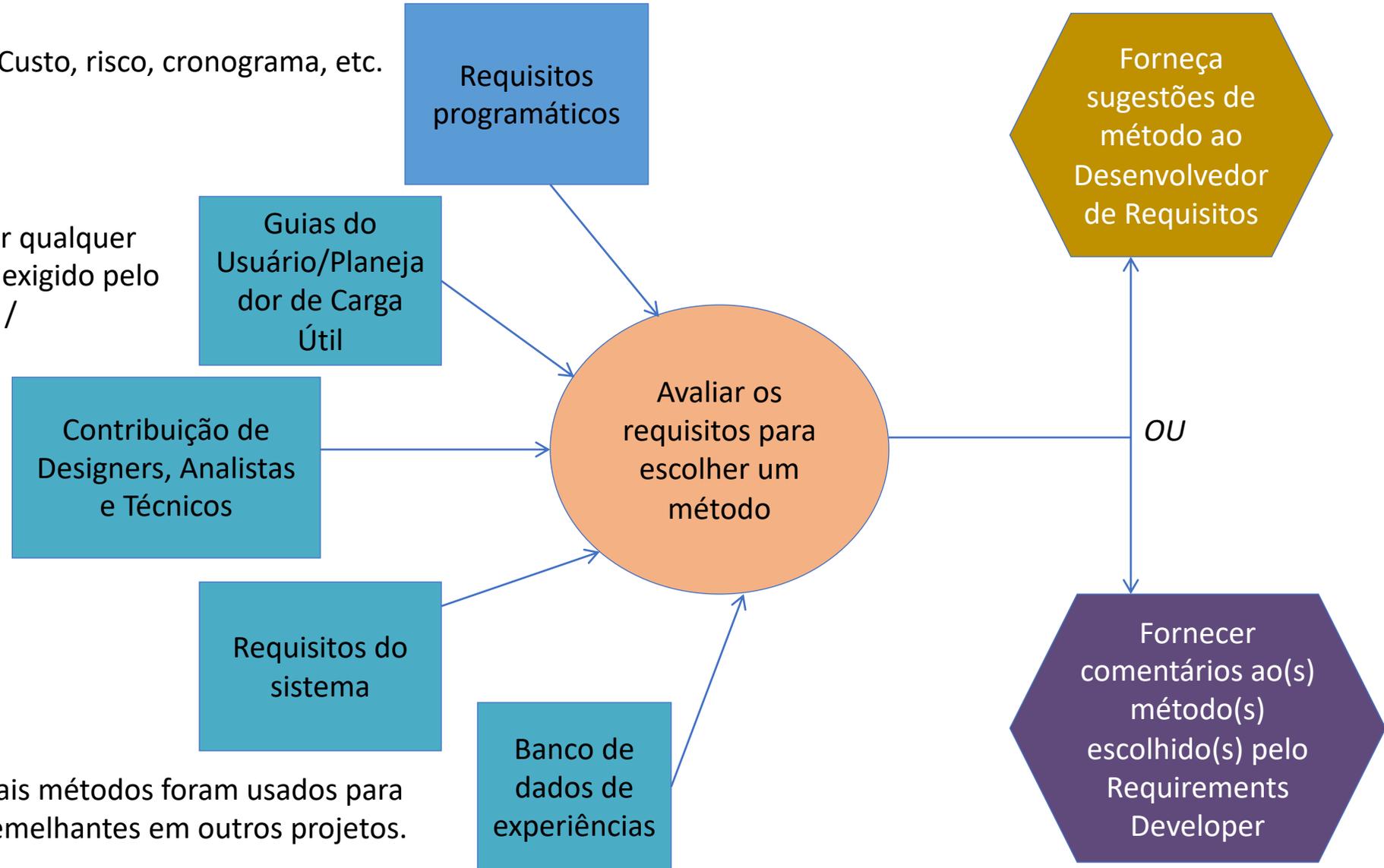


ESCOLHENDO UM MÉTODO V&V

➤ Custo, risco, cronograma, etc.

➤ Revise-os para procurar qualquer método de verificação exigido pelo veículo de lançamento / transportador

➤ Pesquise quais métodos foram usados para requisitos semelhantes em outros projetos.





ESCOLHENDO UM MÉTODO

Teste é usado quando:

- * Técnicas analíticas não produzem resultados adequados
- * Existem modos de falha que comprometem a segurança, os sistemas espaciais ou os objetivos da missão
- * Componentes associados a interfaces críticas do sistema

Demonstração é usada quando:

- * A verificação das funções projetadas pode ser realizada através da observação
- * Os resultados tendem a ser "aprovação/reprovação", "sim/não"
- * Natureza mais subjetiva, como fatores humanos ou capacidade de manutenção
- * Usa sistemas de voo reais ou representativos

A inspeção é usada quando:

- * Desenhos, documentos ou dados podem ser verificados visualmente para verificar se as características físicas foram projetadas no produto
- * Normalmente usado para características de projeto, métodos de construção, mão de obra, dimensões e registros

A análise é usada quando:

- * Técnicas analíticas precisas são possíveis
- * O teste não é uma opção econômica
- * A verificação por inspeção não é adequada
- * Técnicas e modelos analíticos foram validados



- **VR-5T: Prove that the product's communications system is capable of communicating with the ground command team by performing an integrated system test within an integrated hardware/software environment. Testing will show that the product can transmit and receive to standard ground recovery forces audio at frequencies represented by communications devices defined in (TBD).**

Verification Objective

Verification Method

Environment

Note – there are no Special Conditions

Success Criteria



Design Requirement

3.2.2.15.34 Recovery Force Communications
The product shall provide a communications system capable of communicating with the recovery forces pre- and post- landing

Design Verification

Verification Objective	Pass / Fail (Success Criteria)
Perform Integrated System Test of the communications system capability to provide a voice communications and beacon with recovery forces pre and post landing within an integrated hardware / software environment	Testing will show that the communications system can transmit and receive audio at frequencies and ranges (power) represented by standard ground recovery force communications devices as defined in TBD
Perform a demonstration of the communications systems capability to provide voice and beacon communications with recovery forces pre and post landing while within a representative environment and using a production equipment configuration	Demonstration will show the ability for the communications systems to verbally communicate with the on-board communication production configuration equipment. The demonstration will also show beacon tracking within communication ranges established by TBD.

Traceability

Verification Cross Reference Matrix						
Paragraph #	N/A	I	A	M/S	D	T
3.2.2.15.34						VR-5T
3.2.2.15.34					VR-5D	

SE – Translates Operational Objectives into Design Requirements
Design – Provides assessment of requirements implementation
Test – Provides assessment of requirements verifiability

SE – Provides compliance of the design requirement
Test / Implementation Group – Ensures Verification Implementation Feasibility
Advises alternatives to support programmatic
Assesses completeness
Provides verifiability assessment

SE – Verification Allocation and Traceability Assurance



SISTEMAS REATIVOS E MÁQUINAS DE ESTADO



DEFINIÇÃO DE SISTEMAS REATIVOS

- Um sistema reativo é um sistema que, quando ligado, é capaz de criar os efeitos desejados em seu ambiente, **habilitando, aplicando ou impedindo** eventos no ambiente.
- Propriedades:
 - Interação contínua (sem terminação)
 - O sistema responderá a estímulos externos, e
 - a resposta depende do seu estado atual



SISTEMAS TRANSFORMACIONAIS

- Sistemas reativos contrastam com sistemas transformadores, **que existem para transformar uma entrada em uma saída.**
- Propriedades:
 - Apenas para transformar a entrada na saída.
 - Após a transformação, ele termina
 - Não tem estados que sejam relevantes para entidades externas



EXEMPLOS DE SISTEMAS REATIVOS

- **SISTEMAS EM TEMPO REAL** – a resposta depende do tempo. Ex.: software de controle, controle de elevador, caixas eletrônicos
- **SISTEMAS CRÍTICOS** – comportamentos errados podem levar à perda de vidas. Ex.: sistemas biofísicos, computadores de bordo (automóvel/aeronave/nave espacial/navio)
- **SISTEMAS EMBARCADOS** – restrições de implementação. Ex.: móvel, software de firmware integrado, IoT.
- Sistemas que gerenciam infraestruturas críticas: gerenciamento de tráfego aéreo, trem, reatores nucleares, etc.

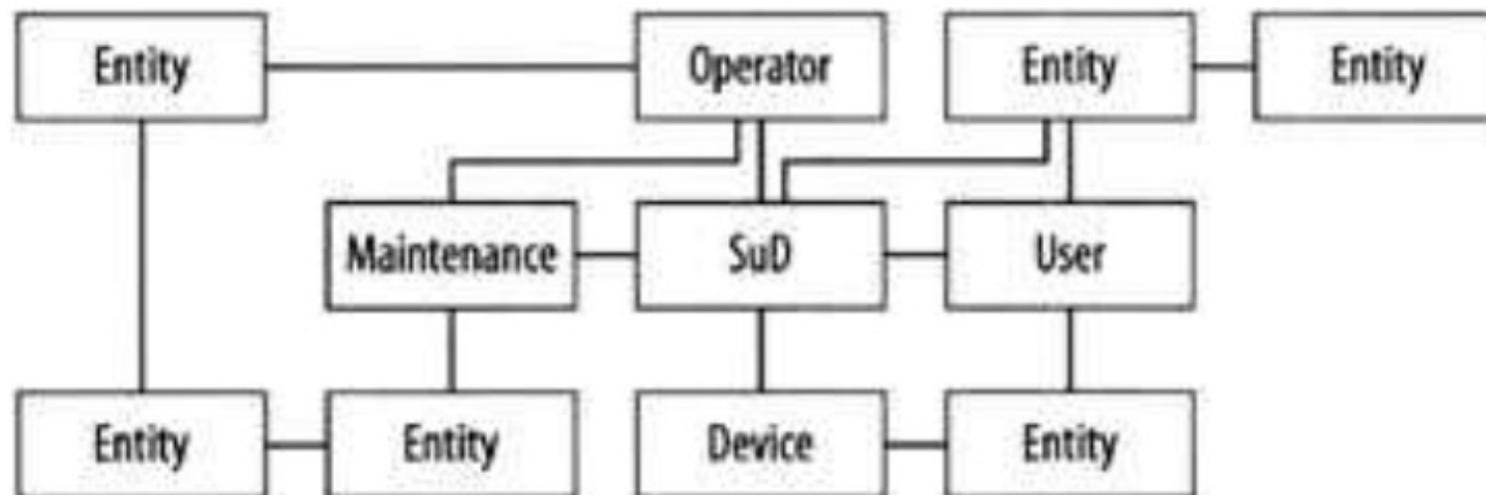


O AMBIENTE



AMBIENTE

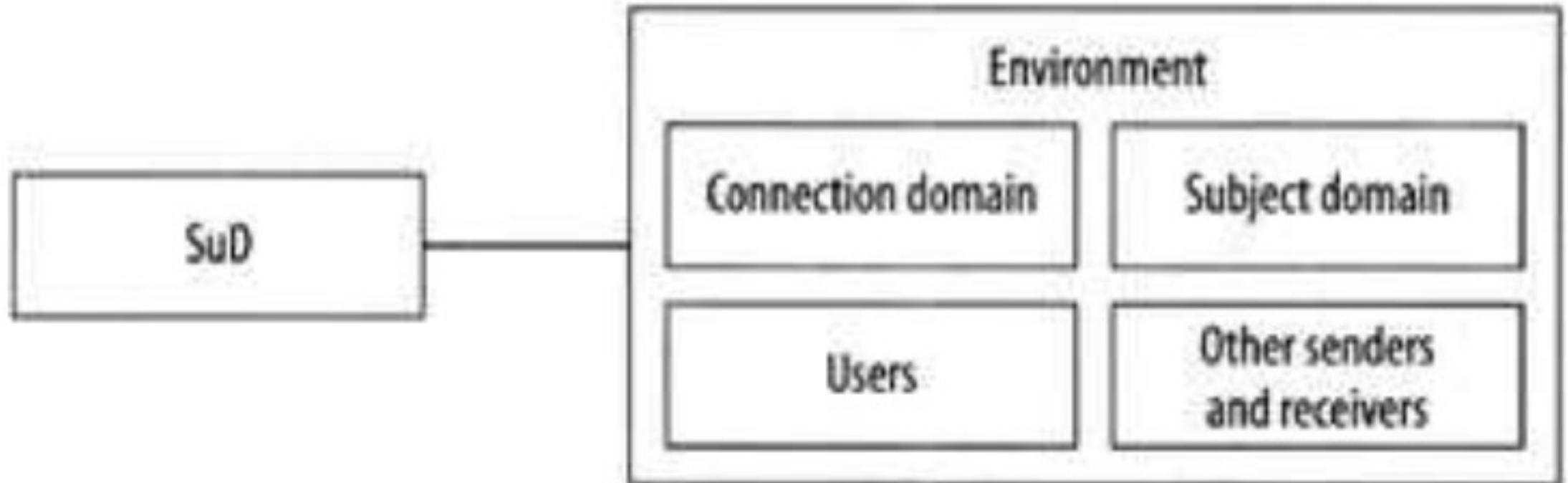
- Precisamos identificar a parte do mundo que é relevante como ambiente do sistema – **Identificar as interfaces do SuD**
 - **As interfaces do sistema podem mudar de acordo com o objetivo ou a fase de teste.**





AMBIENTE

- O ambiente de um sistema contém vários assuntos e domínios conectados.





DOMÍNIO DO ASSUNTO

- DOMÍNIO – domínio do conhecimento que define as dimensões do ambiente (domínio de controle de atitude: roda de reação, magnetômetro, sensor solar, etc.)
 - Entidades de domínio: roda de reação, magnetômetro, etc.
 - Eventos: aplicação de torque, leitura de medição



DOMÍNIO DE CONEXÃO

- Canais de comunicação que permitem a troca de mensagens entre o SuD e o Ambiente.
- Precisamos modelar o canal de comunicação porque isso pode levar a atrasos, dados corrompidos, etc. – **muitos erros acontecem nas interfaces.**

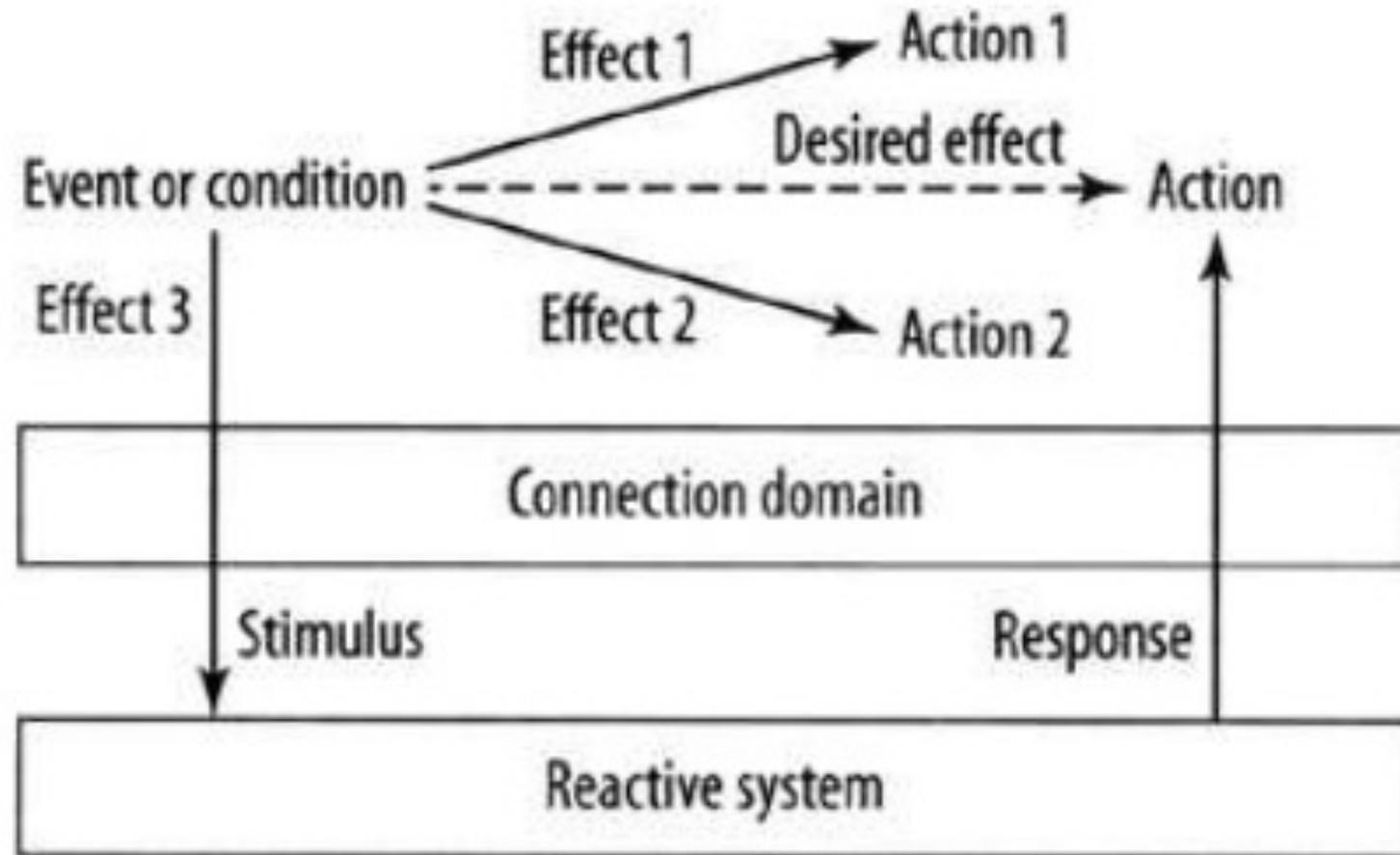


COMPORTAMENTO ESTÍMULO- RESPOSTA



CADEIAS DE CAUSA E EFEITO

- A função de um sistema reativo é **responder à ocorrência de eventos ou condições no ambiente**, causando mudanças no ambiente.





EVENTOS, CONDIÇÕES E AÇÕES

- **EVENTO** – algo que acontece no mundo.
 - **EVENTOS EXTERNOS**: mudança discreta na condição do ambiente
 - **EVENTOS TEMPORAIS**: passagem de um tempo significativo ao qual se espera que o sistema responda
- **CONDIÇÃO** – estado do mundo que persiste por algum período de tempo diferente de zero.
- **AÇÕES** – eventos do ponto de vista do iniciador



ESTÍMULOS

- O estímulo de um sistema é um **evento na interface do sistema causado pelo ambiente**.
- Eventos externos ocorrem em algum lugar do ambiente; os **estímulos ocorrem na interface** do sistema

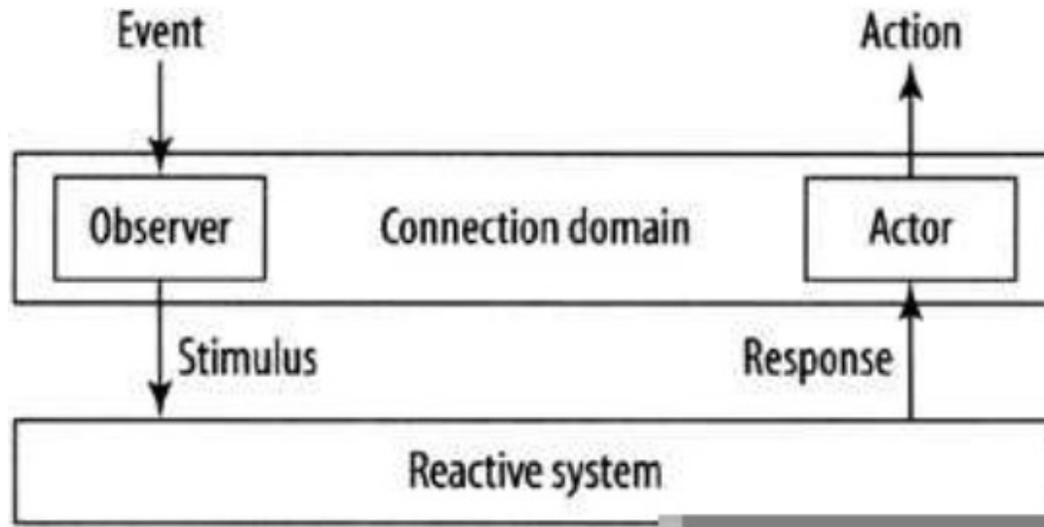


RESPOSTA

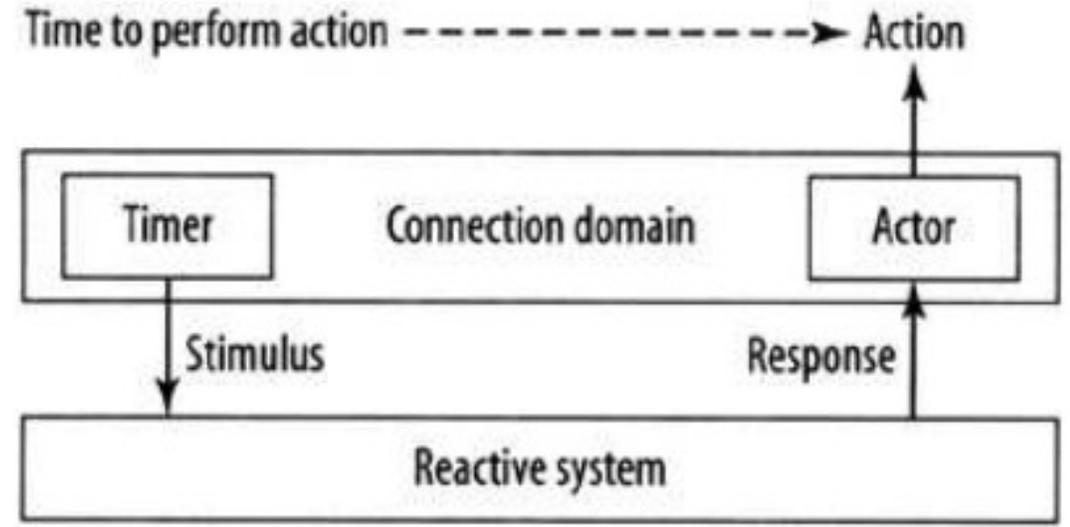
- O sistema reativo **responde** a um evento externo ou temporal atualizando seu estado **ou produzindo uma saída**.
- Uma resposta de saída de um sistema é um **evento na interface** do sistema, causado pelo sistema.
- O efeito desejado do estímulo pode consistir em uma ou mais ações desejadas, a serem causadas por várias respostas.



RESUMO



Eventos-Estímulos



Eventos Temporais



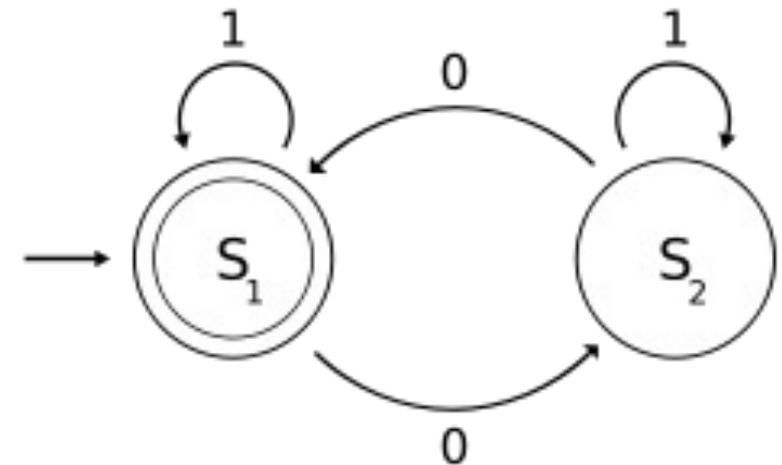
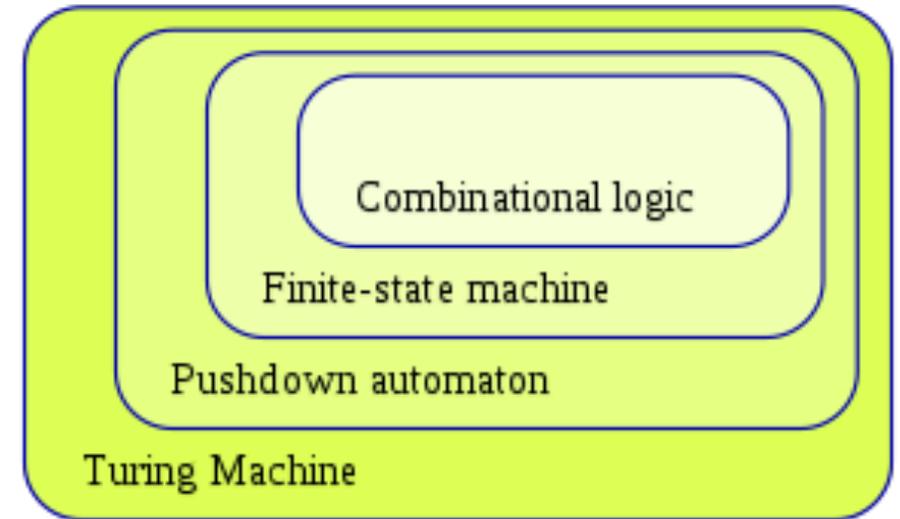
DESCREVENDO COMPORTAMENTO



TEORIA DOS AUTÔMATOS

- Um autômato finito determinístico é representado formalmente por uma quintupla $\langle Q, \Sigma, \delta, q_0, F \rangle$, onde:
- Q é um conjunto finito de estados.
- Σ é um conjunto finito de símbolos, chamado de alfabeto do autômato.
- δ é a função de transição, ou seja, $\delta: Q \times \Sigma \rightarrow Q$.
- q_0 é o estado inicial, ou seja, o estado do autômato antes de qualquer entrada ter sido processada, onde $q_0 \in Q$.
- F é um conjunto de estados de Q (ou seja, $F \subseteq Q$) chamados estados de aceitação.
-

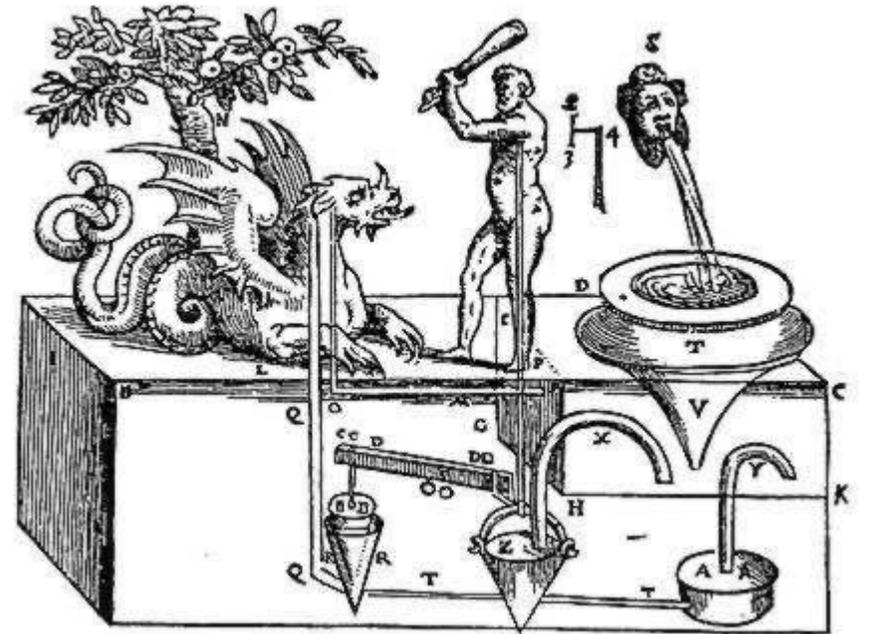
Automata theory





AUTÔMATO

- A própria palavra autômato, intimamente relacionada à palavra "automação", denota processos automáticos que realizam a produção de processos específicos. Simplificando, a teoria dos autômatos lida com a lógica da computação em relação a máquinas simples, referidas como autômatos.
- Autômatos são modelos abstratos de máquinas que executam cálculos em uma entrada movendo-se através de uma série de estados ou configurações. Em cada estado da computação, uma função de transição determina a próxima configuração com base em uma porção finita da configuração atual.





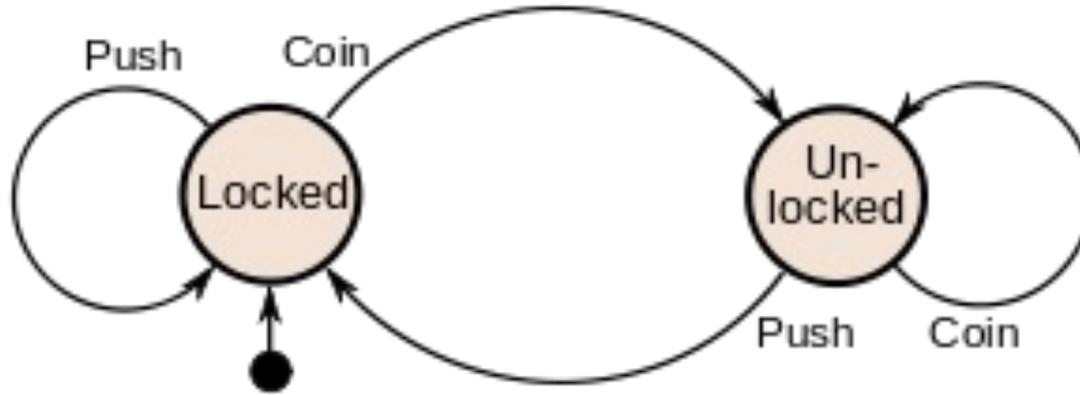
FERRAMENTA: MÁQUINAS DE ESTADO

- Uma máquina de estados finitos (FSM) ou autômato de estados finitos (FSA, plural: autômatos), autômato finito, ou simplesmente uma máquina de estados, é um modelo matemático de computação.
- **É uma máquina abstrata que pode estar em exatamente um de um número finito de estados a qualquer momento.**
- O FSM pode **mudar de um estado para outro** em resposta a algumas entradas externas; a mudança de um estado para outro é chamada de **transição**.
- Uma FSM é definida por uma lista de seus estados, seu estado inicial e as condições para cada transição.





EXEMPLO



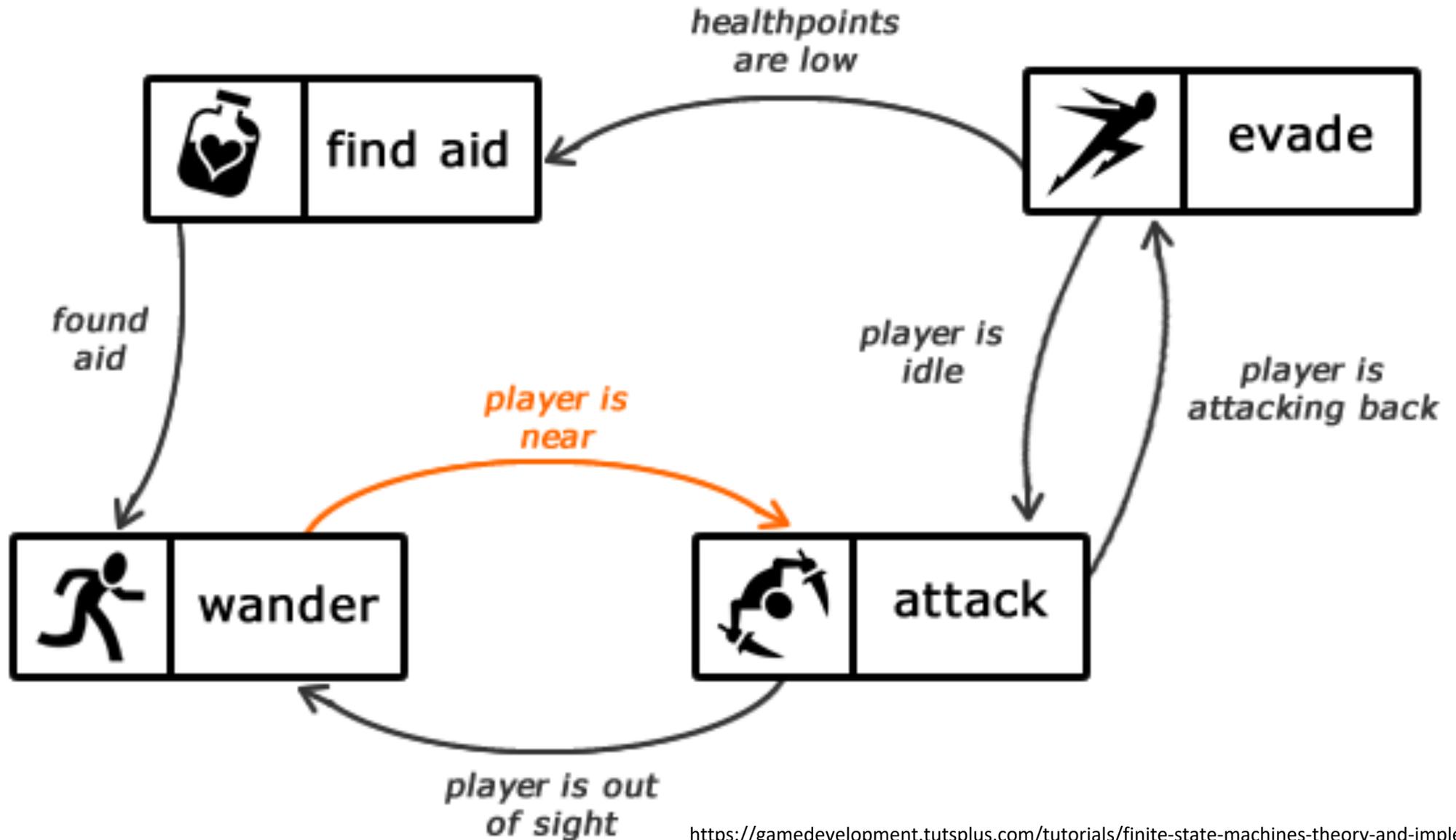
Current State	Input	Next State	Output
Locked	coin	Unlocked	Unlocks the turnstile so that the customer can push through.
	push	Locked	None
Unlocked	coin	Unlocked	None
	push	Locked	When the customer has pushed through, locks the turnstile.





UM EXEMPLO DE JOGO

FSM representando o cérebro de um inimigo.





O DIAGRAMA DA MÁQUINA DE ESTADO

<https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-state-machine-diagram/>

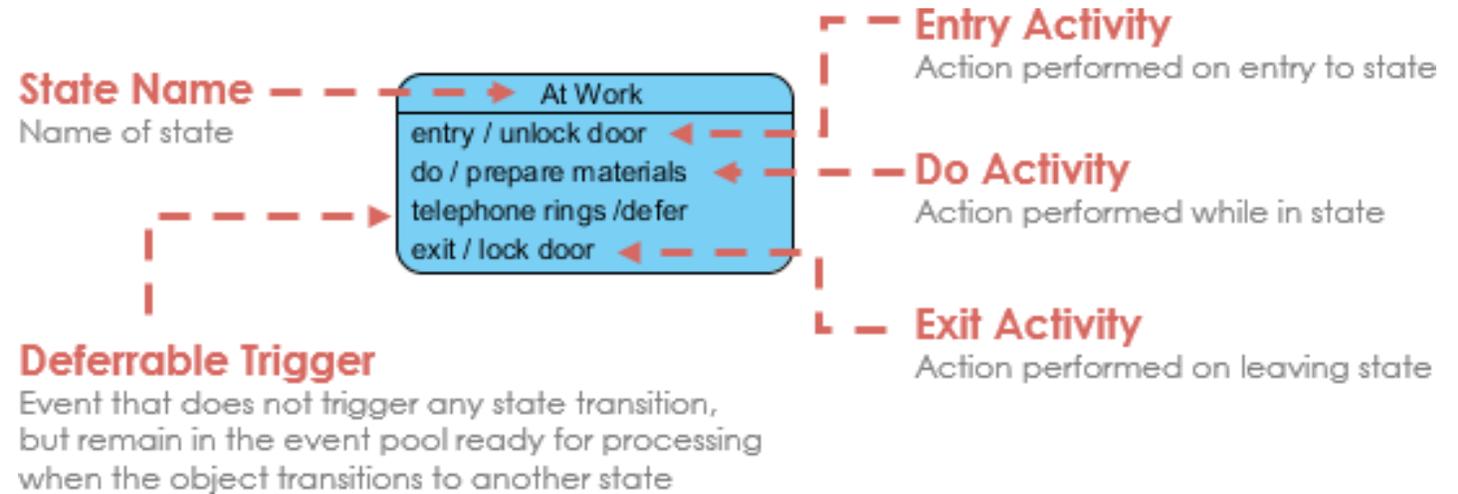
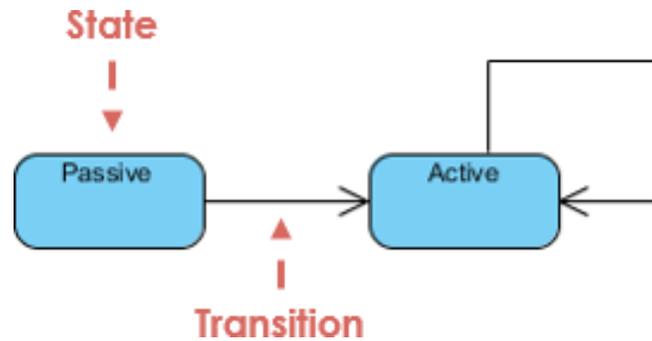


VISÃO GERAL

- O comportamento de uma entidade não é apenas uma consequência direta de suas entradas, mas também depende de seu estado anterior. A **história passada de uma entidade pode ser melhor modelada** por um diagrama de máquina de estados finitos ou tradicionalmente chamado de autômatos.
- Os Diagramas de Máquina de Estado (ou às vezes chamados de diagrama de estado, máquina de estado ou gráfico de estado) mostram os diferentes estados de uma entidade.
- Os diagramas de máquina de estado também **podem mostrar como uma entidade responde a vários eventos** mudando de um estado para outro.



ESTADO

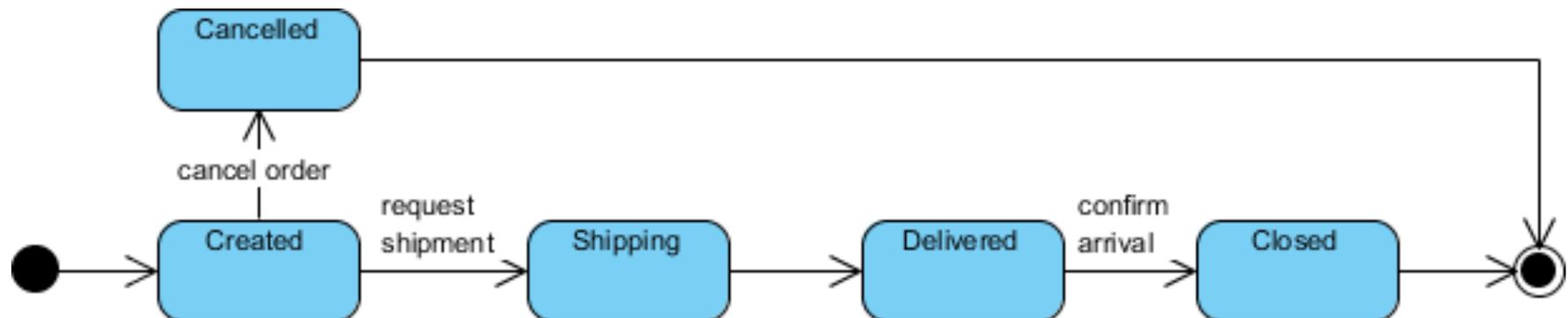


- Um estado é uma restrição ou uma situação no ciclo de vida de um objeto, na qual uma restrição se mantém, o objeto executa uma atividade ou aguarda um evento.
- Existem várias características dos estados em geral, independentemente de seus tipos:
 - Estado representa as condições dos objetos em determinados pontos no tempo.
 - Um estado é frequentemente associado a uma abstração de valores de atributo de uma entidade que satisfaz alguma(s) condição(ões).
 - Uma entidade muda seu estado não apenas como consequência direta da entrada atual, mas também depende de algum histórico passado de suas entradas.



ESTADOS INICIAIS E FINAIS

- O estado inicial de um diagrama de máquina de estados, conhecido como **pseudo-estado inicial**, é indicado com um círculo sólido. **Uma transição deste estado mostrará o primeiro estado real**
- O **estado final** de um diagrama de máquina de estado é mostrado como círculos concêntricos. Uma máquina de estado de **loop aberto representa um objeto que pode terminar antes que o sistema termine**, enquanto um diagrama de máquina de estado **de loop fechado não tem um estado final**; se for o caso, então o objeto vive até que todo o sistema termine.





TRANSIÇÃO

- **As linhas de transição retratam o movimento de um estado para outro.** Cada linha de transição é rotulada com o **evento que causa a transição**.
 - Compreender as transições de estado faz parte da análise e do projeto do Sistema
- Várias transições ocorrem quando eventos diferentes resultam no encerramento de um estado ou quando há condições de proteção (guarda) nas transições
- **Uma transição sem um evento e uma ação é conhecida como transições automáticas**



O DIAGRAMA DA MÁQUINA DE ESTADO DO SYSML

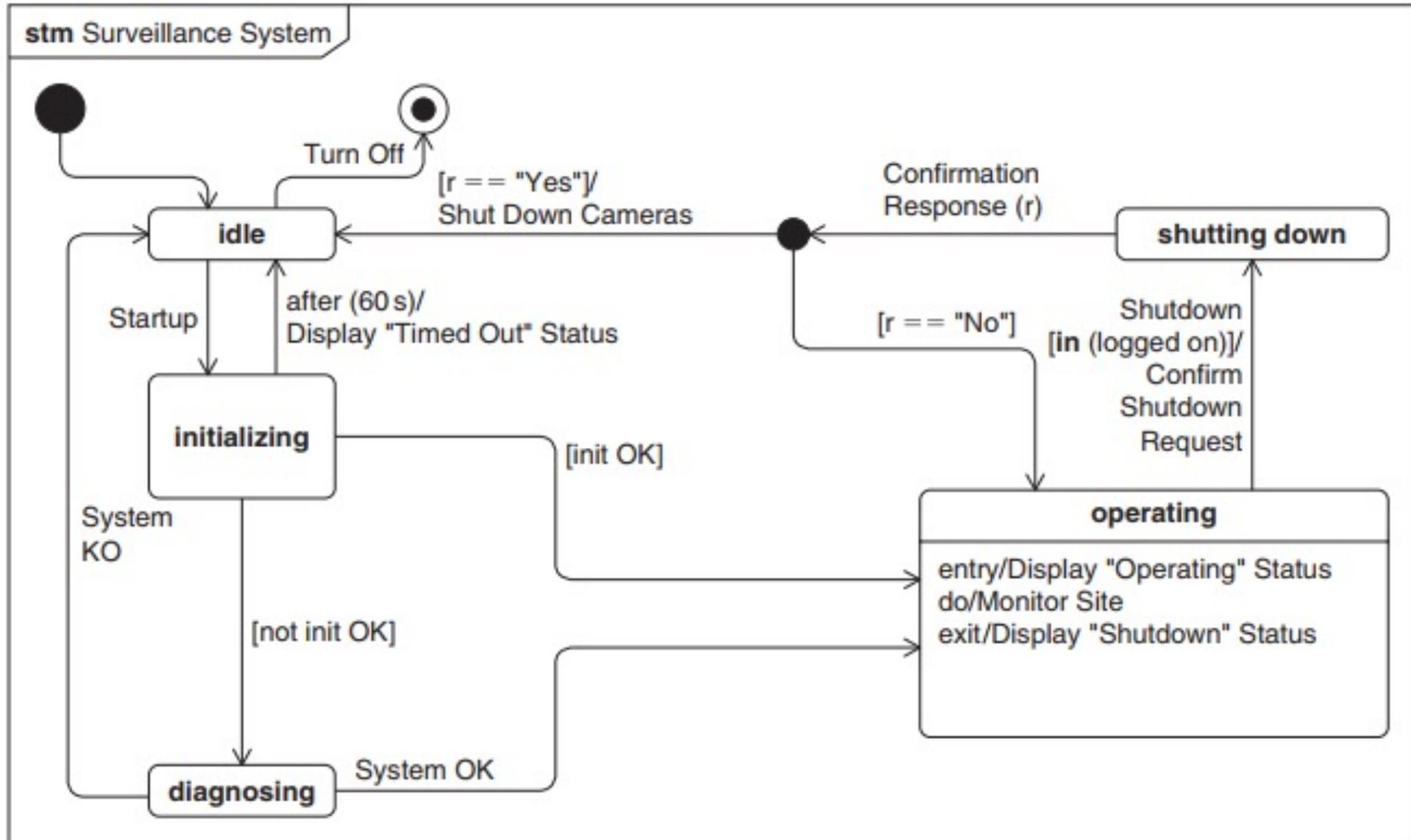


VISÃO GERAL

- As máquinas de estado são normalmente usadas para descrever o comportamento dependente do estado de um elemento da arquitetura ao longo de seu ciclo de vida, que é definido em termos de seus estados e as transições entre eles..
- A máquina de estado define como o comportamento do elemento muda à medida que ele transita entre diferentes estados e enquanto está em diferentes estados.
- As máquinas de estado podem ser usadas para descrever uma ampla gama de comportamentos relacionados à estados, desde um simples interruptor de lâmpada até os modos complexos de uma aeronave avançada.



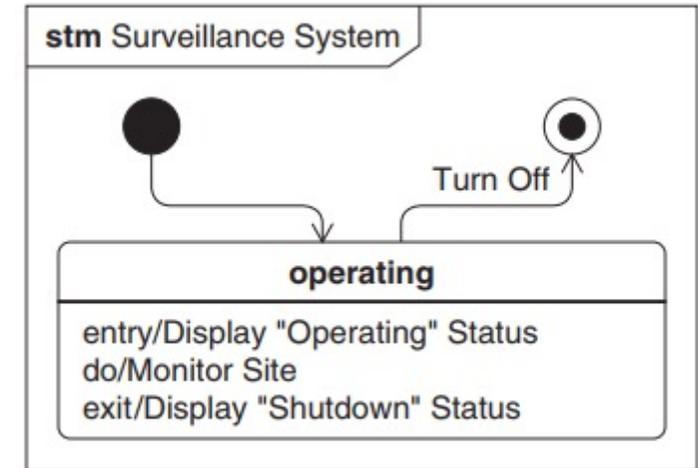
EXEMPLO DE MÁQUINA DE ESTADO





ESPECIFICANDO ESTADOS

- **REGIÃO:** descrever o comportamento relacionado ao estado da máquina de estado.
- **ESTADO:** representa alguma condição significativa na vida de um elemento, normalmente porque representa alguma mudança na forma como responde aos eventos e quais comportamentos ele executa.
 - Cada estado pode conter comportamentos de entrada e saída que são executados sempre que o estado é inserido ou encerrado, respectivamente. Além disso, o estado pode conter um comportamento que é executado uma vez que o comportamento de entrada tenha sido concluído





TRANSIÇÕES DE ESTADO

- Uma transição específica quando uma mudança de estado ocorre dentro de uma máquina de estado.
 - Trigger (Gatilho)
 - Guard (Guarda)
 - Effect (Efeito)



GATILHO

- Um gatilho identifica os possíveis estímulos que causam a ocorrência de uma transição. O SysML tem quatro tipos principais de eventos de disparo.
 - Um **evento de sinal** indica que uma nova mensagem assíncrona correspondente a um sinal chegou. Um evento de sinal pode ser acompanhado por vários argumentos que podem ser usados no efeito de transição.
 - Um **evento de tempo** indica que um determinado intervalo de tempo passou desde que o estado atual foi inserido (relativo) ou que um determinado instante no tempo foi atingido (absoluto).
 - Um **evento de alteração (mudança)** indica que alguma condição foi satisfeita (normalmente que algum conjunto específico de valores de atributo mantém).
 - Um **evento de chamada** indica que uma operação no bloco proprietário da máquina de estado foi solicitada. Um evento de chamada também pode ser acompanhado por vários argumentos.



GUARDA

- A guarda de transição contém uma expressão que deve ser avaliada como verdadeira para que o efeito de uma transição ocorra ou a própria transição. A guarda é especificada usando uma restrição, que inclui uma expressão textual para representar a condição de guarda.
- Quando um evento satisfaz um gatilho, a proteção na transição é avaliada.
 - Se for avaliado como verdadeiro, a transição será acionada;
 - se for avaliado como falso, o evento será consumido sem efeito.

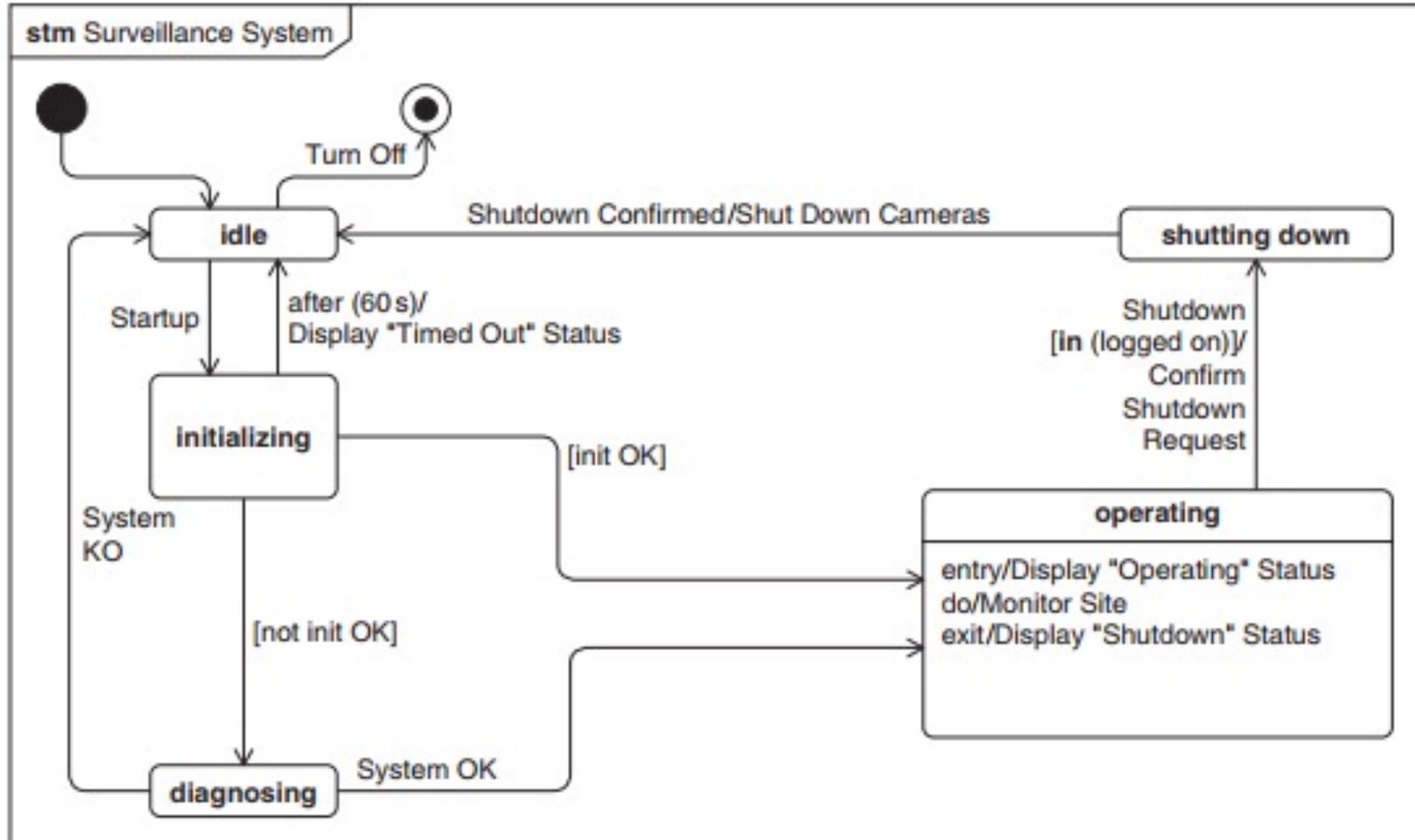


EFEITO

- O efeito é um **comportamento executado durante a transição** de um estado para outro.
- O efeito de transição pode ser um comportamento arbitrariamente complexo que pode incluir ações de sinal de envio ou chamadas de operação usadas para interagir com outros elementos.
 - Se a transição for acionada, primeiro o comportamento de saída do estado atual (origem) será executado, o efeito de transição será executado e, finalmente, o comportamento de entrada do estado de destino será executado.



TRANSIÇÕES





PSEUDOESTADOS: INICIALIZAÇÃO E CONCLUSÃO

- A inicialização e a conclusão de uma região são descritas usando um pseudoestado inicial e um estado final, respectivamente.
- Um **pseudoestado inicial** é usado para determinar o estado inicial de uma região.
- Quando o estado ativo de uma região é o estado final, a região foi concluída e não ocorrem mais transições dentro dela. Assim, um **estado final não pode ter transições de saída**.
- O pseudoestado **“terminate”** é sempre associado ao estado de uma máquina de estado inteira.
 - Se um pseudoestado de terminação for atingido, o comportamento da máquina de estado será encerrado. Um pseudoestado de terminação tem o mesmo efeito que atingir os estados finais de todas as regiões da máquina de estado.

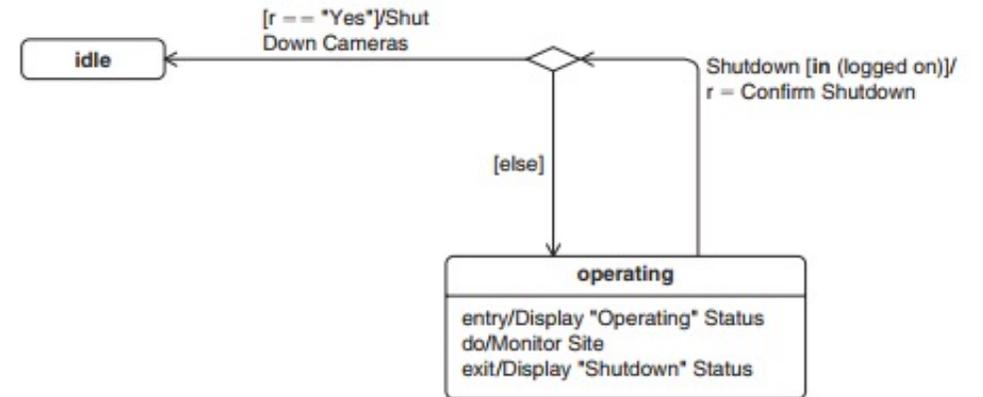
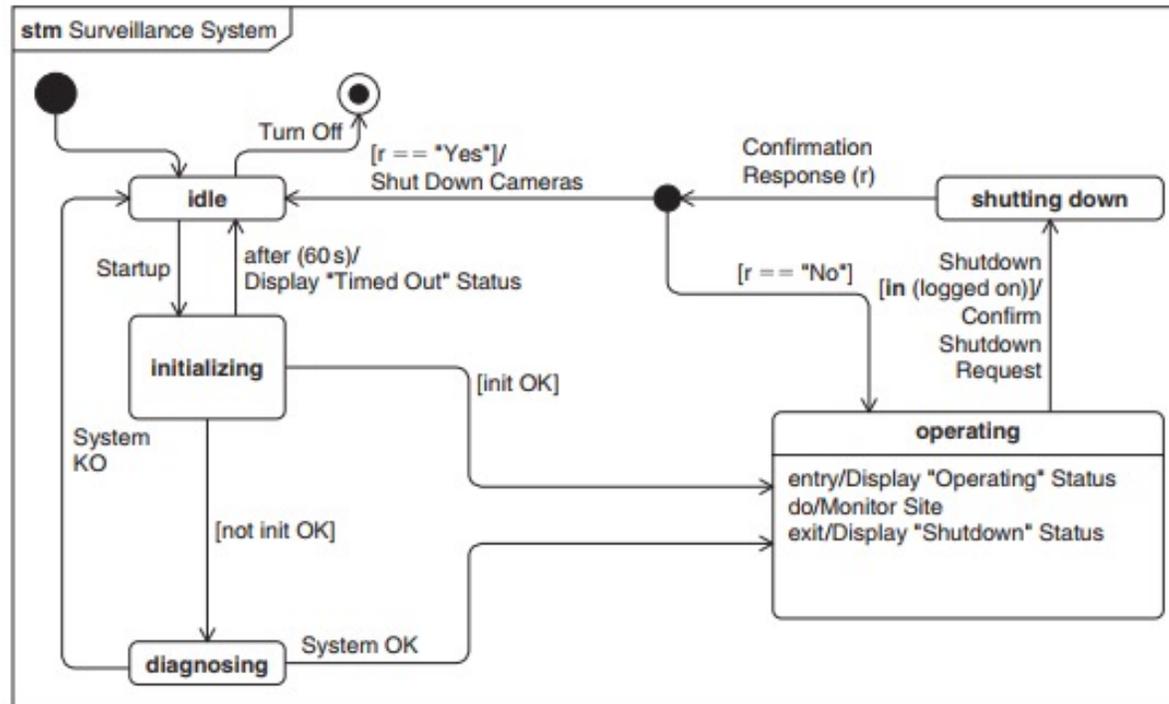


TRANSIÇÕES DE ROTEAMENTO USANDO PSEUDOESTADOS

- Há uma variedade de situações em que uma simples transição direta entre dois estados não é suficiente para expressar a semântica necessária..
 - Um **pseudoestado de junção** é usado para construir um caminho de transição composto entre estados. A transição composta **permite que mais de um caminho de transição alternativo entre estados seja especificado**, embora apenas um caminho possa ser tomado em resposta a qualquer evento único..
 - O **pseudoestado de escolha** também tem múltiplas transições de entrada e transições de saída e, como o pseudoestado de junção, é parte de uma transição composta entre estados. O comportamento do pseudoestado de escolha é distinto do de um pseudoestado de junção na medida em que **os guardas em suas transições de saída não são avaliados até que o pseudoestado de escolha tenha sido atingido**.



ROUTINGS



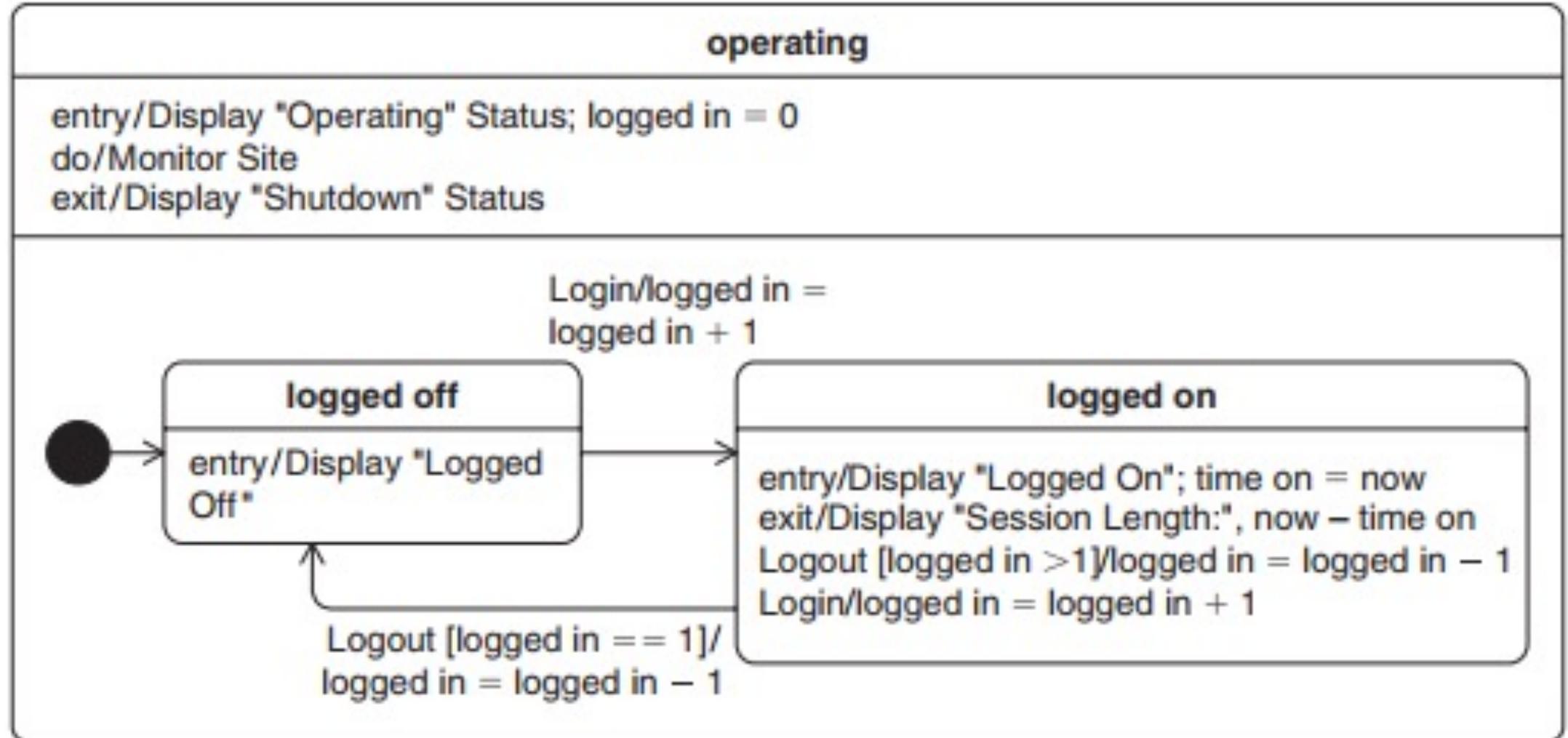


ESTADOS COMPOSTOS – REGIÃO ÚNICA

- **Indiscutivelmente, a situação mais comum é um estado composto que tem uma única região.**
 - Uma região normalmente conterá um pseudoestado inicial e um estado final, um conjunto de pseudoestados e um conjunto de subestados, que podem ser estados compostos..
 - Se a região tiver um estado final, um evento de conclusão será gerado quando esse estado for atingido.
- Um estado composto pode ter muitas regiões, que podem conter subestados.



ESTADOS COMPOSTOS – REGIÃO ÚNICA (DOIS SUBESTADOS)



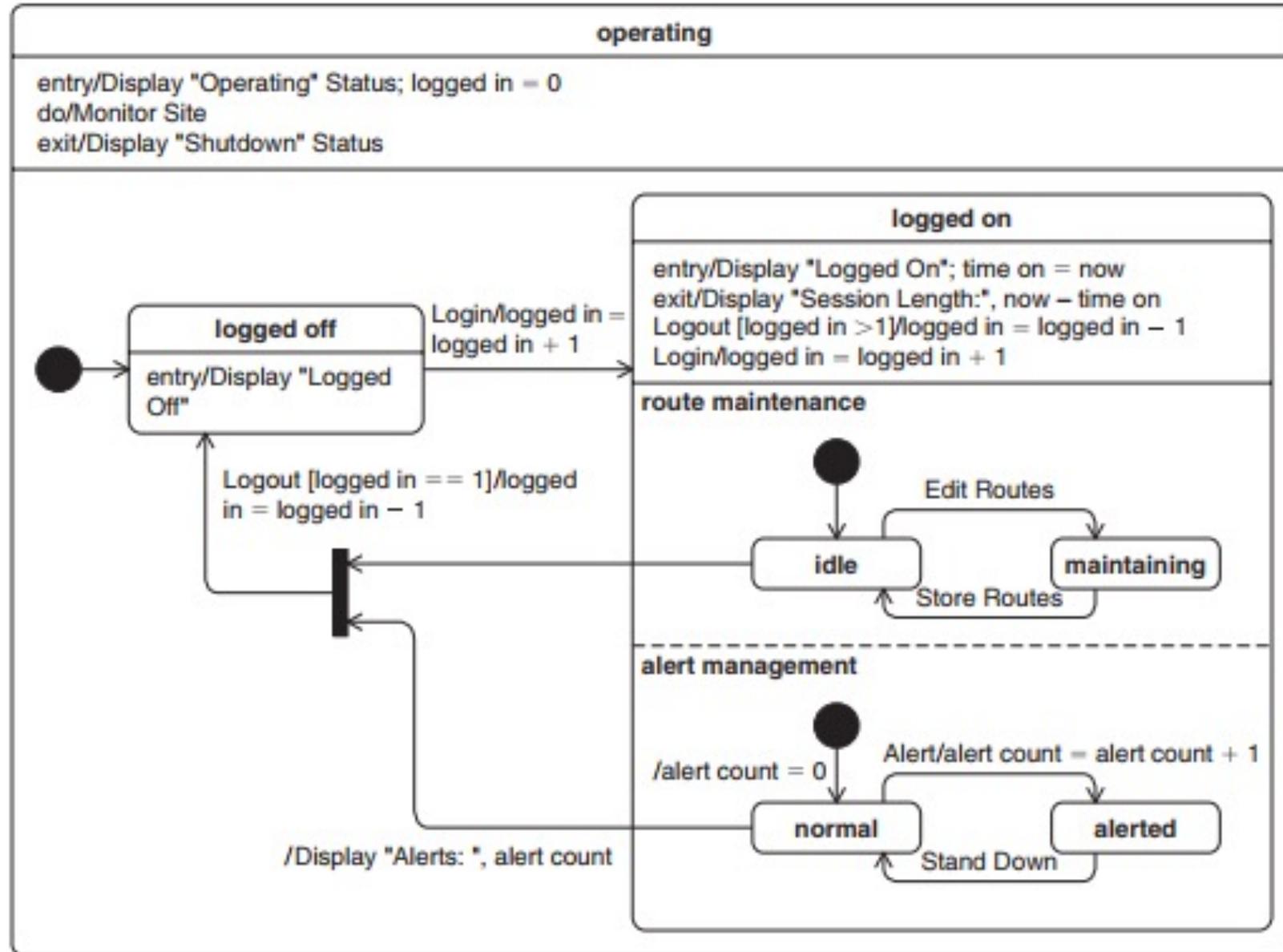


ESTADOS COMPOSTOS - SIMULTANEIDADE

- Quando um **estado composto ortogonal** está ativo, cada região tem **seu próprio estado ativo** que é independente das outras, e **qualquer evento de entrada é analisado independentemente** dentro de cada região.
 - Uma transição que termina no estado composto **acionará transições do pseudoestado inicial** de cada região, portanto, deve haver um pseudoestado inicial em cada região para que tal transição seja válida.
- Da mesma forma, um **evento de conclusão para o estado composto ocorrerá quando todas as regiões estiverem em seu estado final.**



ESTADOS COMPOSTOS - SIMULTANEIDADE





PSEUDOESTADO DA HISTÓRIA

- Em alguns cenários de design, é desejável manipular um evento de exceção interrompendo o comportamento da região atual, respondendo ao evento e, em seguida, retornando ao estado em que a região estava no momento da interrupção.
 - Um **pseudoestado de história profunda** registra os estados de todas as regiões na hierarquia de estado abaixo e incluindo a região que possui o pseudoestado de história profunda.
 - Um **pseudoestado de histórico raso** registra apenas o estado de nível superior da região que o possui.



CONSIDERAÇÕES FINAIS



Table A.21 State Machine Diagram State Nodes

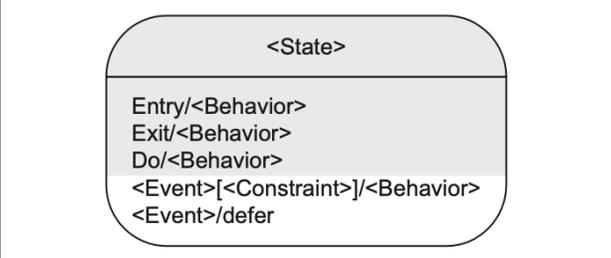
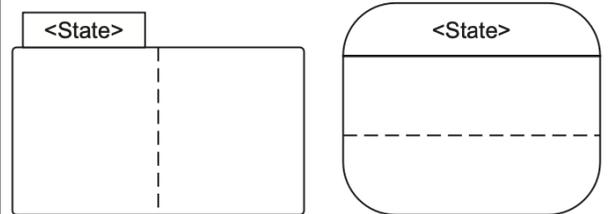
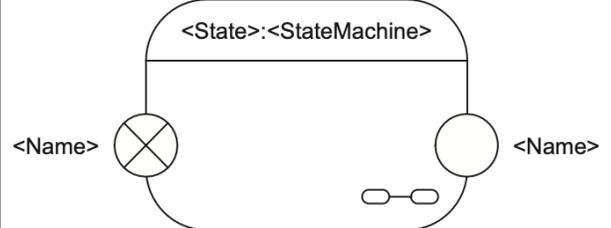
Diagram Element	Notation	Description	Section
State Machine with Entry- and Exit-Point Pseudostate Nodes		<p>A state machine may have entry- and exit-point pseudostates, which are similar to junctions. On state machines, entry-point pseudostates can only have outgoing transitions and exit-point pseudostates can only have incoming transitions.</p>	11.6.5
Atomic State Node		<p>A state represents some significant condition in the life of a block. Each state may have entry and exit behaviors, and a do behavior.</p> <p>An atomic state node may also show transitions that are local to the state and events that are deferred while the state machine is in this state.</p>	11.3
Composite State with Entry- and Exit-Point Pseudostate Nodes		<p>A composite state is a state with nested regions; the most common case is a single region.</p> <p>A composite state may have entry- and exit-point pseudostates that act like junction pseudostates. Entry points have incoming transitions from outside the state and exit points have the opposite.</p>	11.6.1
Composite State Node with Multiple Regions		<p>A composite state may have many regions, which may each contain substates. These regions are orthogonal to each other and so a composite state with more than one region is sometimes called an orthogonal composite state.</p>	11.6.2
Sub-State Machine Node with Connection Points		<p>A state machine may be reused using a kind of state called a submachine state. A transition ending on a submachine state will start its referenced state machine. Transitions may also be connected to connection points on the boundary of the state. The symbol in the lower right refers to a lower level statemachine diagram. This can also be represented by a rake symbol.</p>	11.6.5



Table A.22 State Machine Diagram Pseudostate and Transition Nodes

Diagram Element	Notation	Description	Section
Terminate Pseudostate Node		If a terminate pseudostate is reached, then the behavior of the state machine terminates.	11.3
Initial Pseudostate Node		An initial pseudostate specifies the initial state of a region.	11.3
Final State Node		The final state indicates that a region has completed execution.	11.3
Choice Pseudostate Node		The outgoing transitions of a choice pseudostate are evaluated once it has been reached.	11.4.2
Junction Pseudostate Node		A junction pseudostate is used to construct a compound transition path between states.	11.4.2
Trigger Node		This node represents all the transition's triggers, with the descriptions of the triggering events and the transition guard inside the symbol.	11.4.3
Action Node		<EffectExpression> describes the effect of the transition, either the name of a behavior or the body of an opaque behavior.	11.4.3
Send Signal Node		This node represents a send signal action. The signal's name, together with any arguments that are being sent, are shown within the symbol.	11.4.3
Join Pseudostate Node		A join pseudostate has a single outgoing transition and many incoming transitions. When all of the incoming transitions can be taken, and the join's outgoing transition is valid, then all the transitions happen.	11.6.2
Fork Pseudostate Node		A fork pseudostate has a single incoming transition and many outgoing transitions. When an incoming transition is taken to the fork pseudostate, all of the outgoing transitions are taken.	11.6.2
History Pseudostate Node		A history pseudostate represents the last state of its owning region, and a transition ending on a history pseudostate has the effect of returning the region to the state it was last in.	11.6.4



Table A.23 State Machine Diagram Paths

Diagram Element	Notation	Description	Section
Time Event Transition Path	$\begin{array}{l} \text{after } \langle \text{TimeExpression} \rangle [\langle \text{Constraint} \rangle] / \langle \text{Behavior} \rangle \\ \hline \longrightarrow \\ \text{at } \langle \text{TimeExpression} \rangle [\langle \text{Constraint} \rangle] / \langle \text{Behavior} \rangle \\ \hline \longrightarrow \end{array}$	Time events indicate either that a given time interval has passed since the current state was entered (after), or that a given instant of time has been reached (at). The transition can also include a guard and effect.	11.4.1
Signal Event Transition Path	$\begin{array}{l} \langle \text{Signal} \rangle (\langle \text{Attribute} \rangle, \dots) [\langle \text{Constraint} \rangle] / \langle \text{Behavior} \rangle \\ \hline \longrightarrow \end{array}$	Signal events indicate that a new asynchronous message has arrived. A signal event may be accompanied by a number of arguments, which may be assigned to attributes. The transition can also include a guard and effect.	11.4.1
Call Event Transition Path	$\begin{array}{l} \langle \text{Operation} \rangle (\langle \text{Attribute} \rangle, \dots) [\langle \text{Constraint} \rangle] / \langle \text{Behavior} \rangle \\ \hline \longrightarrow \end{array}$	Call events indicate that an operation on the state machine's owning block has been requested. A call event may also be accompanied by a number of arguments, which may be assigned to attributes. The transition can also include a guard and effect.	11.5
Change Event Transition Path	$\begin{array}{l} \text{when } \langle \text{Expression} \rangle [\langle \text{Constraint} \rangle] / \langle \text{Behavior} \rangle \\ \hline \longrightarrow \end{array}$	Change events indicate that some condition has been satisfied (normally that some specific set of attribute values hold). The transition can also include a guard and behavior/effect.	11.7



ROTEIRO DOS DIAGRAMAS

