Controle supervisório de robô manipulador em célula de manufatura de dois processos

Disciplina de Controle para Automação
Dept. Engenharia Elétrica da Universidade de Brasília

Carlos - 14/0036920, Iracema Mendes - 14/0042091,
Ricardo Caldas - 12/0045800, Thalbert Miranda- 14/0057056

***Abstract*— O trabalho final da disciplina de controle para automação[1] propõe que o grupo modele, por meio de autômatos, o comportamento do robô manipulador *Cyton* em célula de manufatura com dois processos distintos, e conceba três controladores supervisórios baseados nas especificaçõees apresentadas. O grupo modelou cada um dos processos da planta localmente, e por meio de composição paralela chegou ao modelo geral da planta. Este por sua vez, foi composto com cada uma das especificações para gerar os três controles supervisórios, que passaram por processo de verificação de modelo no UPPAAL e posteriormente implementados e testados na planta física.**

I. FUNDAMENTAÇÃO TEÓRICA

O referêncial teórico utilizado neste projeto se baseia no livro usado na disciplina *Discrete Event Systems*[1] e notas de aula providas pelo professor. Nesta seção apresentamos os dois conceitos mais importantes empregados no projeto.

A. Autômatos

Autômato é uma ferramenta capaz de representar uma linguagem de acordo com regras bem definidas. O autômato finito determinístico G é definido pela tupla

$$G = (X, E, f, \Gamma, x_0, X_m) \tag{1}$$

onde

X é o conjunto finito de estados,
E é o conjunto finito de eventos associados a G,
$f : X \text{ x } E \rightarrow X$ é a função de transição: $f(x, e) = y$ que significa que em um estado x a ocorrência de e o leva para y,
$\Gamma : X \rightarrow 2^E$ é a função de eventos ativos; $\Gamma(x)$ é o conjunto de todos os eventos e para cada $f(x, e)$ que está definido,
x_0 é o estado inicial, e
$X_m \subseteq X$ são os estados marcados.

B. Composição paralela

Composição paralela, ou composição síncrona, é a operação de união entre autômatos que atuam como processos concorrentes. A composição paralela dos autômatos G_1 e G_2 é dada pelo autômato

$$G_1||G_2 := Ac(X_1 \text{x} X_2, E_1 \cup E_2, f, \Gamma_{1||2}, (x_{01}, x_{02}), X_{m1} \text{x} X_{m2}) \tag{2}$$

Portanto, um evento comum, que pertence a ambos os autômatos, somente pode ser executado caso esteja definido para os dois estados compostos. E por isso os autômatos são chamados sincronizados por eventos.

II. SISTEMA FÍSICO

O sistema físico onde foram realizados os experimentos consiste do braço robótico Cyton entre duas máquinas que realizam os processamentos G e R nas peças. Estas, de cores verde e vermelho, ficam disponíveis em um buffer de entrada, no qual o sistema é capaz de identificar a cor e enviar um comando ao sistema informando a chegada, por meio dos eventos não-controláveis ('new*'). O Cyton então executa movimentos de carregar ou descarregar as máquinas, em que carregar ('*Load/red2Green') move a peça para uma das máquinas e descarregar ('*Unload') retira a peça das máquinas e as coloca no final da célula para recolhimento.

III. MODELAGEM

A modelagem foi feita na ferramenta de verfificação de modelos formais UPPAAL, porém com a formalização de autômatos aprendida ao decorrer do curso de Controle para Automação. Contudo, nota-se duas diferenças inevitáveis: o estado inicial é marcado por dos círculos concêntricos e os estados marcados (nas especificações e supervisórios) são coloridos de acordo com a ocupação das máquinas verde e vermelho.

Modelamos localmente os processos básicos do sistema e por composição paralela destes obtivemos o modelo geral da planta. Em seguida, as especificações foram modeladas individualmente seguindo as respectivas descrições. Finalmente a composição paralela do modelo geral com cada uma das especificações gerou os controladores supervisórios específicos a cada uma.

A. Modelos locais do sistema

Cyton: A Figura 3a detalha o autômato que modela o robô cyton na planta.

[1]Repositório do Projeto Desenvolvido Disponível em https://github.com/rdinizcal/CPA-022017

Processo G: A Figura 3b detalha o autômato que modela o processo da máquina G na planta.
Processo R: A Figura 3c detalha o autômato que modela o processo da máquina R na planta.

B. Modelo geral do sistema

A Figura 3e detalha o autômato que modela o processo geral da planta, dado pela composição paralela dos modelos locais do sistema.

C. Modelo dos supervisórios

- Supervisório 1
  A Figura 5 representa o supervisório do modelo do sistema geral que obedece à especificação 1 do roteiro do trabalho, sendo dado pela composição paralela do modelo do sistema geral e da especificação 1, fig. 4a.
- Supervisório 2
  A Figura 6 representa o supervisório do modelo do sistema geral que obedece à especificação 2 do roteiro do trabalho, sendo dado pela composição paralela do modelo do sistema geral e da especificação 2, fig. 4c.
- Supervisório 3
  A Figura 7 representa o supervisório do modelo do sistema geral que obedece à especificação 3 do roteiro do trabalho, sendo dado pela composição paralela do modelo do sistema geral e da especificação 3, fig. 4b.

IV. IMPLEMENTAÇÃO

A implementação foi feita em cima do middleware ROS à partir da arquitetura de referência provida pelos monitores implementando os estados modelados para os 3 supervisórios. A partir dos códigos disponibilizados pelo monitor foi possível o desenvolvimento de 3 novos códigos relativos a cada um dos autômatos supervisórios modelados. Os códigos possuem o seguinte padrão:

Para cada estado do autômato foi criada uma função em python, sendo o nome da função o nome do estado correspondente, ilustrado na Figura 1.

```
def C0G0R0E11():#1
        print('state1(C0G0R0E11)')

        global state
        global newEvent

        while True:
                if newEvent == 'greenLoad':
                        sendMovement(1)#greenLoad!
                        newEvent = 'none'
                        state = C1G1R0E10
                        break
```

Fig. 1: Exemplo de função

Ao iniciar o estado um *print* é realizado de forma a facilitar a visualização de que estado o robô se encontra. Em seguida é necessário chamar as variáveis globais 'state' e 'newEvent' que correspondem ao estado e ao novo evento respectivamente. Uma vez dentro do estado é necessário aguardar um novo evento e para isso um *loop* 'while True' é criado e ele tem como condição de saída o sistema receber um novo evento correspondente ao de mudança de estado daquele estado específico. Após verificar o novo evento quaisquer movimentos a serem realizados pelo robô são feitos utilizando a função 'sendMovement()' que recebe como parâmetros:

```
sendMovement(0): "redLoad", busca peça no buffer de entrada e deixa no processo "R";
sendMovement(1): "greenLoad", busca peça no buffer de entrada e deixa no processo "L";
sendMovement(2): "redUnload", busca peça no processo "R" e deixa no buffer de saída;
sendMovement(3): "greenUnload", busca peça no processo "G" e deixa no buffer de saída;
sendMovement(4): "red2Green", busca peça no processo "R" e deixa no processo "G"
```

Fig. 2: Parâmetros da função sendMovement()

Por fim a variável 'newEvent' é definida para 'none' de forma a voltar a ficar esperando por um novo evento, e a variável 'state' é definida como sendo o próximo estado a ser executado a partir do evento recebido e então o *loop* é quebrado e a função encerrada. Em sequência o código executa a próxima função cujo nome é o definido na variável 'state'.

V. RESULTADOS

O teste da implementação se deu em um modelo físico disposto no Laboratório de Robótica Aplicada (LARA) da Universidade de Brasília. Os resultados obtidos para a primeira e terceira especificação foram satisfatórios, uma vez que dada uma sequência de comandos a resposta do robô foi condizente com o previsto. Para a segunda especificação não foi encontrado o resultado desejado, isto ocorreu pois o código não estava de acordo com o modelo projetado, uma vez que foi encontrado um estado em que o robô deveria esperar um comando, mas no código a espera era por um comando diferente, prejudicando o desempenho nesta etapa.

VI. CONCLUSÕES

O projeto e implementação do presente trabalho abordam todo o processo de automação de uma célula de manufatura com dois processos por meio do controle de um robô manipulador. Além de exercitar a técnica de controle supervisório para sistemas a eventos discretos possibilitando a automação de processos com o intuito de otimização, segurança e robustez.

Este se mostrou extremamente relevante no entendimento da capacidade de desenvolvimento de projeto de controle para automação de processos fabris, o que proporcionou aos alunos envolvidos uma experiência valiosa para a formação em Engenharia de Controle e Automação. Já que fizeram parte do processo inteiro, desde a concepção do modelo formal do sistema até o projeto do controle supevisório para modelos de sistemas discretos e finalmente implementação e validação no protótipo físico do LARA.

REFERENCES

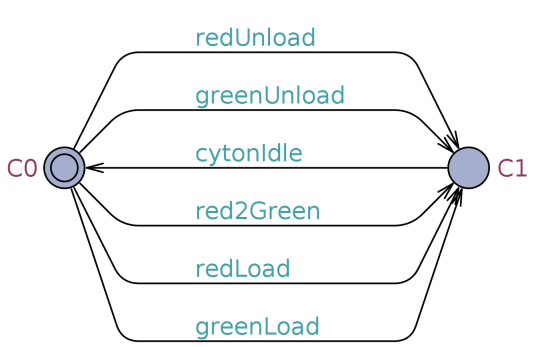
[1] Christos G. Cassandras and Stephane Lafortune. *Introduction to Discrete Event Systems*. Springer Publishing Company, Incorporated, 2nd edition, 2010.

Apêndice

A. Modelos da Planta

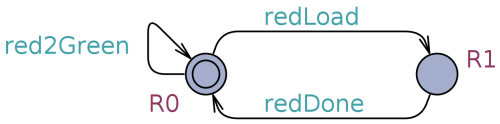
B. Especificações

C. Supervisórios



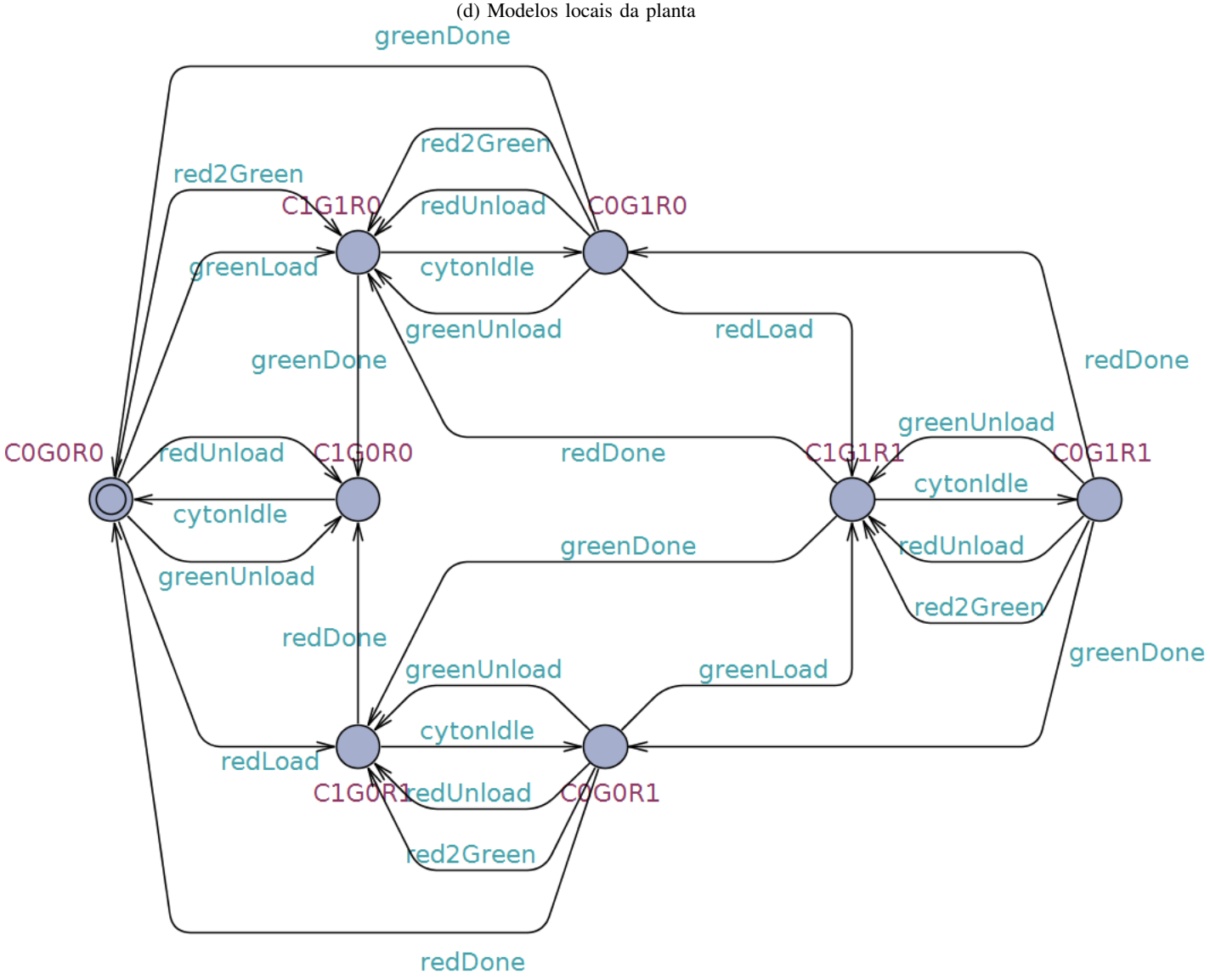
(a) Modelo do robô Cyton

(b) Modelo do processo G



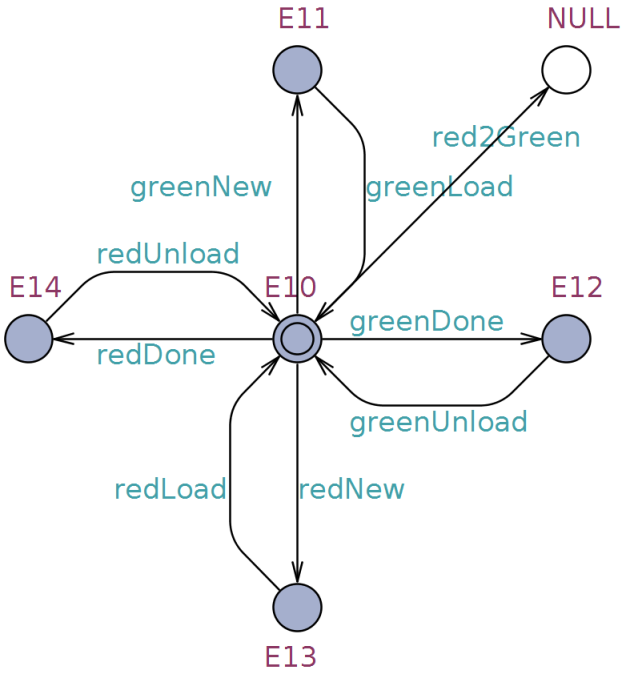
(c) Modelo do processo R

(d) Modelos locais da planta

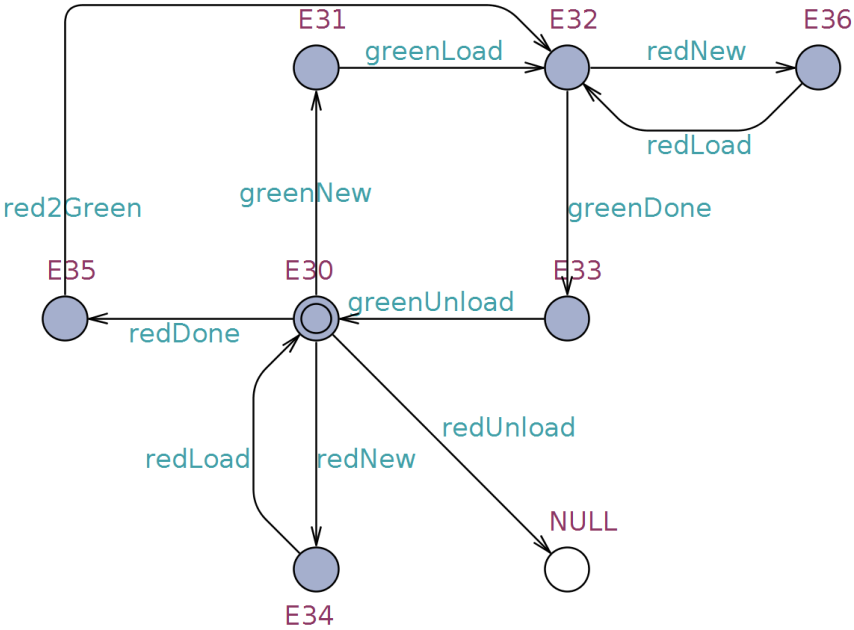


(e) Modelo global da planta

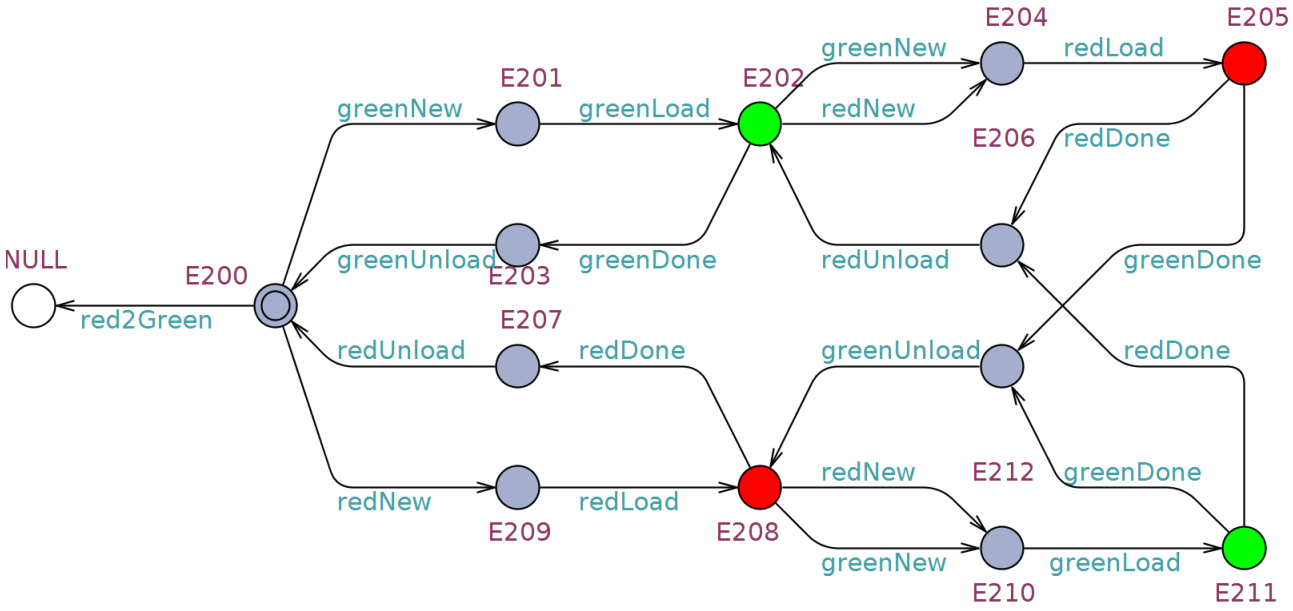
Fig. 3: Modelos da Planta



(a) Especificação 1



(b) Especificação 3



(c) Especificação 2

Fig. 4: Especificações

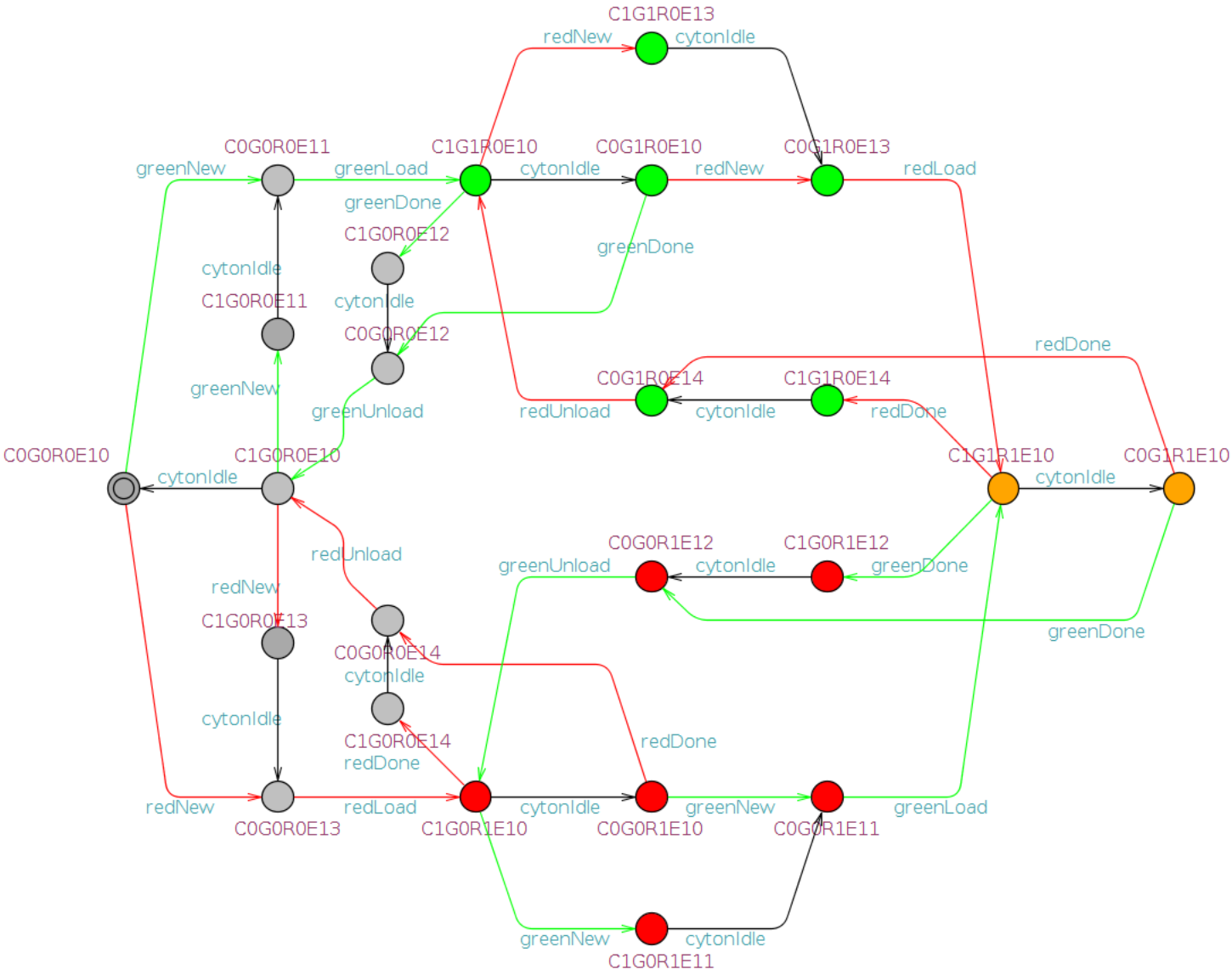


Fig. 5: Supervisório 1

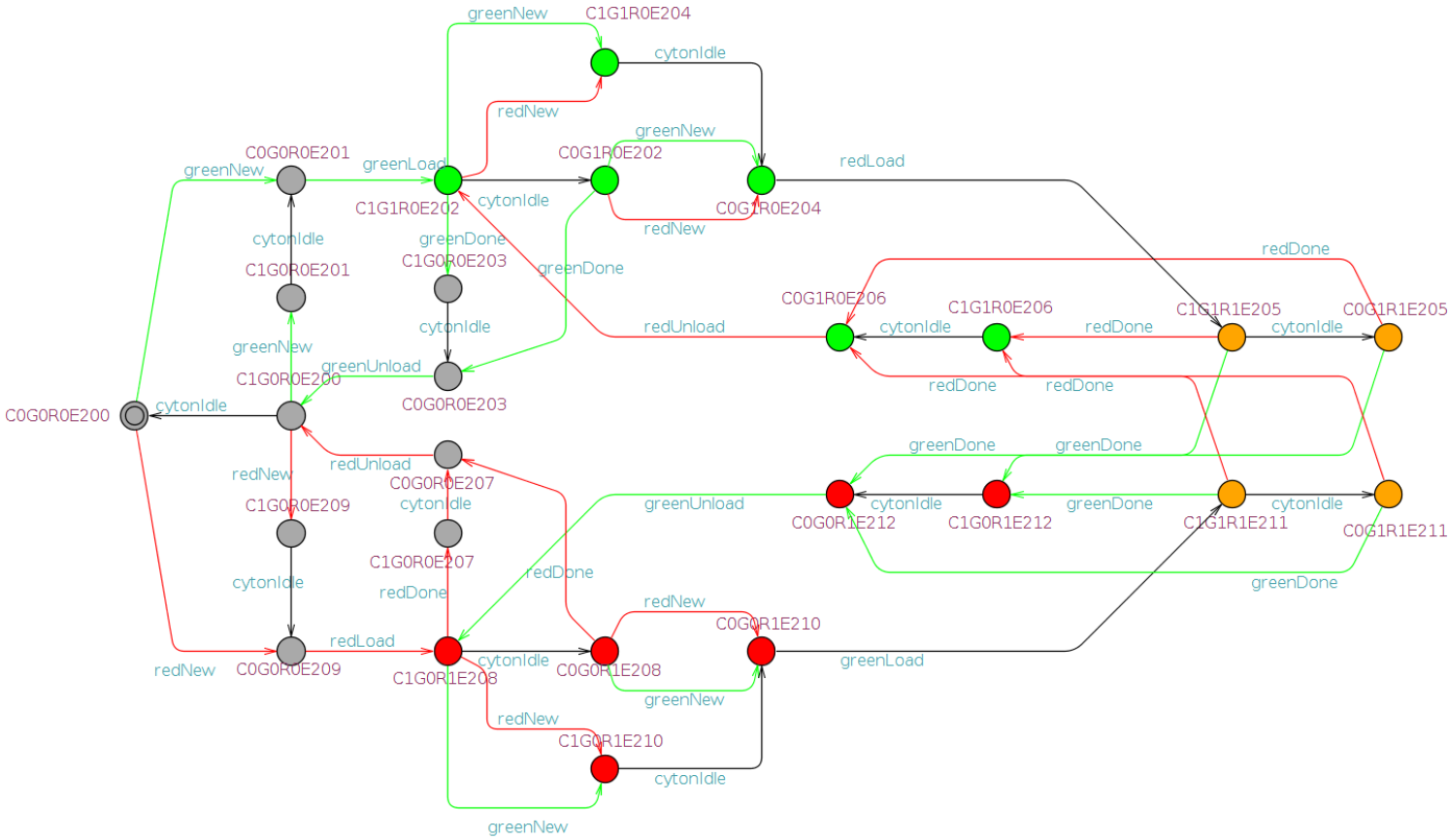


Fig. 6: Supervisório 2

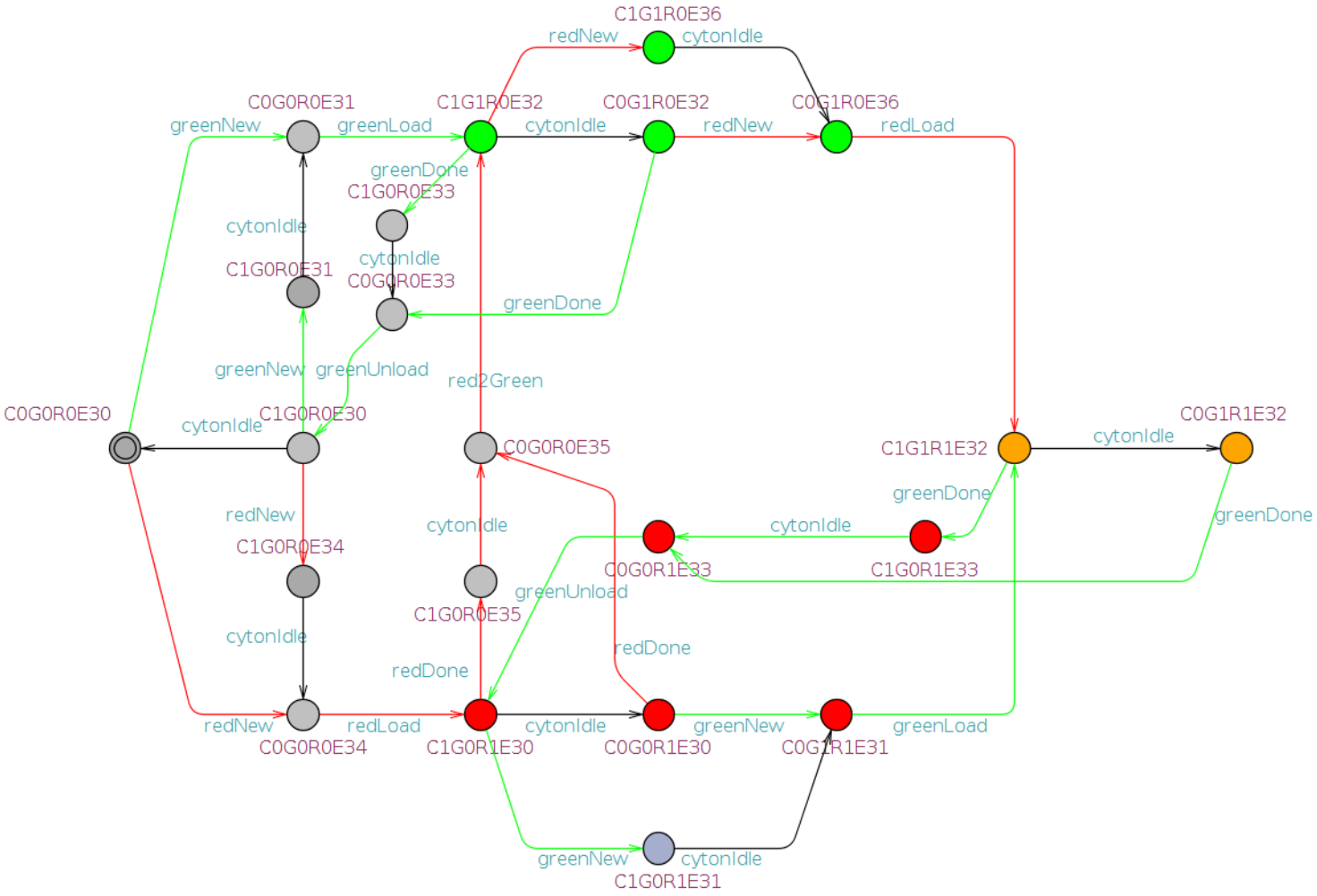


Fig. 7: Supervisório 3