Presentation On “Transforming Flowchart Into Coloured Petri Nets”

Kaaviyashri Saraboji

Details of the paper

Title of the paper: Transforming Flowcharts into Colored Petri Nets

Authors: Utumporn Gulati and Wiwat Vatanawood

Published in: Proceedings of the 2019 International Conference on Software and e-Business (ICSEB)

- ***Location & Date:*** Tokyo, Japan & December 9-11, 2019
- ***Pages****:* 75-80

Publisher*:* Association for Computing Machinery (ACM)

Objectives of the paper

To address the lack of tools for validating flowchart logic before implementation.

To propose a formal transformation scheme that converts flowcharts into Colored Petri Nets(CPN).

To define a set of mapping rules that guide the transformation from flowchart elements to CPN components.

To enable automatic simulation and formal verification of flowchart logic using CPN Tools.

Structural representation: Flowchart vs Colored Petri Net

Flowchart Structure

- Flowchart is a graphical representation of a process or step-by-step solution used in early-stage of software design
- Represents control flow using: Start, Process, Decision, Connector, End nodes.
- Easy to Understand, but lacks simulation and formal verification
- Cannot model data flow or validate logic automatically

Colored Petri Net Structure

- Formal model used for system simulation and verification.
- Composed of Places(hold tokens), Transitions(represent actions), Arcs(connect places and transitions)
- Tokens carry data values defined by color sets (e.g., INT, BOOL, STRING)
- Supports formal analysis and logic validation using CPN Tools

Mapping Rules: Flowchart to CPN Transformation

Flowchart Element	CPN Representation
Start Node	One Place + One Transition (initial token)
Process Node	One Place + One Transition (with arc inscription)
Decision Node	One Transition with Guard
Connector Node	One Place + One Transition + One Arc
End Node	One Place (final token destination)

Transformation Workflow: Import, Extraction, and Mapping

Step 1 – Import XML Flowchart:
- Flowchart is prepared in a structured XML format
- This format enables automated parsing and element identification.

Step 2 – Extract Elements:
- Flowchart elements such as Start, Process, Decision, Connector, End nodes are extracted.
- Each element includes an ID, type, and control flow relationships

Step 3 – Apply Mapping Rules:
- Extracted node is transformed into corresponding CPN components
- The transformation follows the mapping rules.

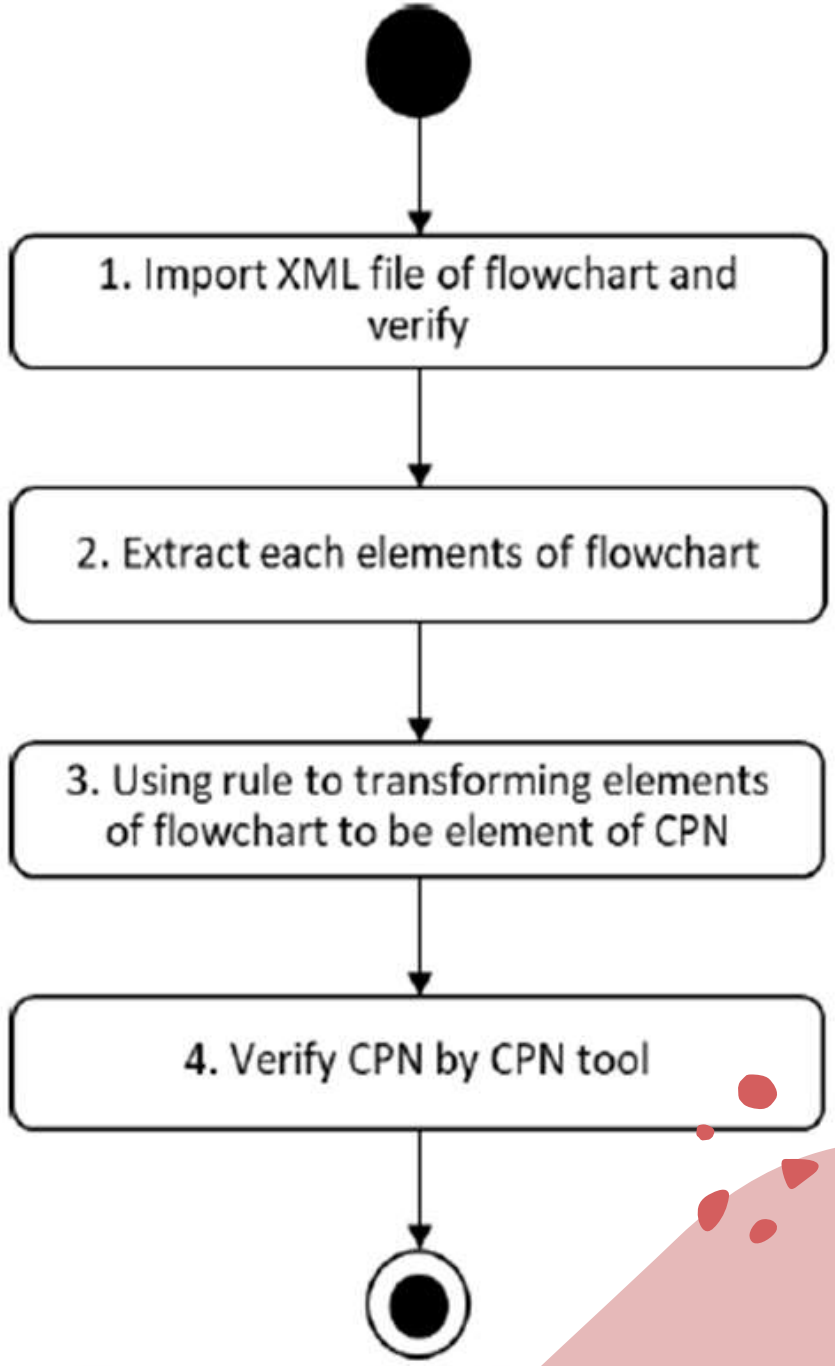


Figure 4. The overview of our T

Variable declaration and Data Handling in CPN

- Flowchart variable (e.g., `score`, `grade`) are modeled as tokens in CPN
- Tokens are assigned data types using color sets:

  `colset INT = int;` integer values (e.g., `85`)

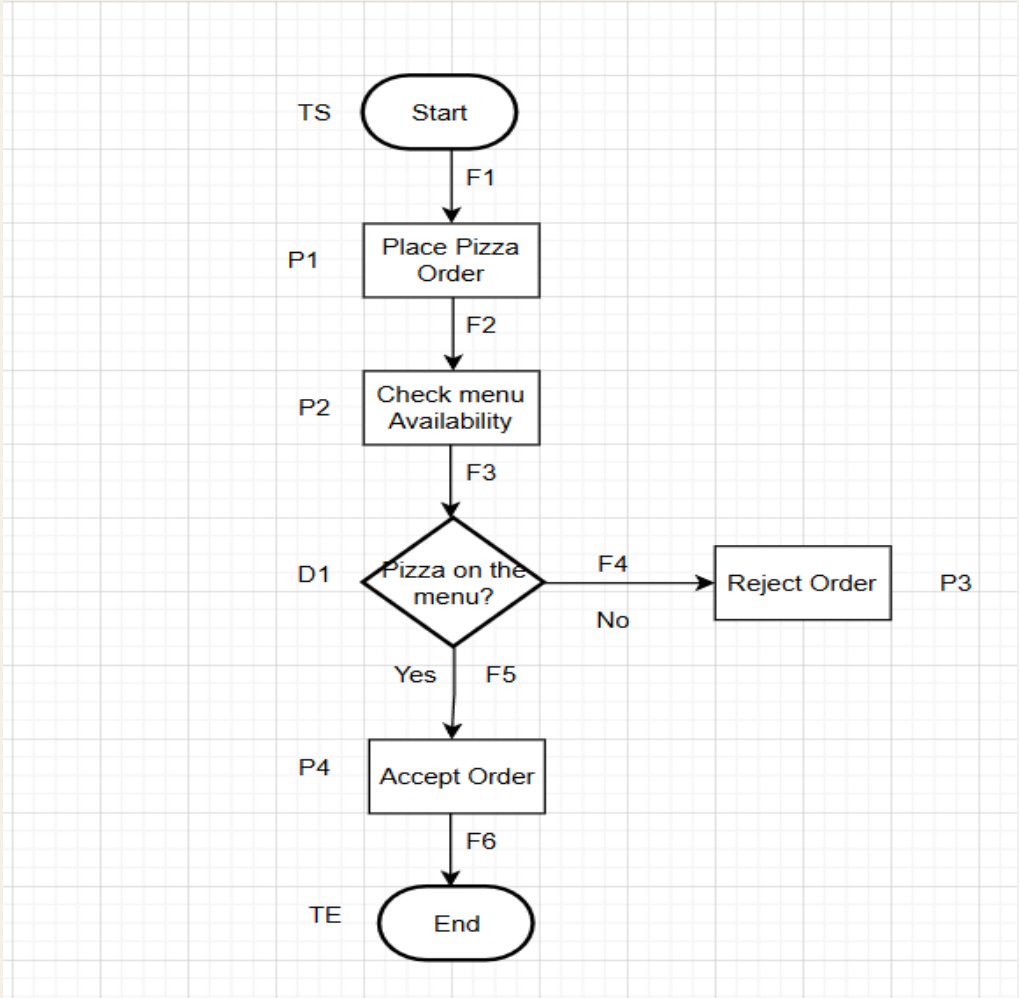
  `colset BOOL = bool;` logical conditions (e.g., `true`)

  `colset STRING = string;` text (e.g., `"A"`, `"Fail"`)
- Variables are declared using `var` and are linked to places (where token resides) or arcs (where data flows)
- Arc inscriptions specify: How tokens are read from input places, How values are updated or passed during transitions
- Initial markings define the starting value of a variable/token. (e.g., var score : INT; init 85;)

Simulation and Validation using CPN Tools

- After transformation, the CPN model is loaded into CPN Tools.
- Simulation is used to observe token movement and system behavior
- Transitions fire when input places contain valid tokens and guard conditions are met
- Tokens are updated and passed to output places, simulating flowchart logic
- Simulation verifies:
  - **Correct decision logic**
  - **Proper data updates** (e.g., score → grade)
  - **Whether the system terminates correctly** (token reaches end place)
- Enables formal validation of flowchart logic before implementation

My flowchart to CPN transformation and verification using CPN Tools

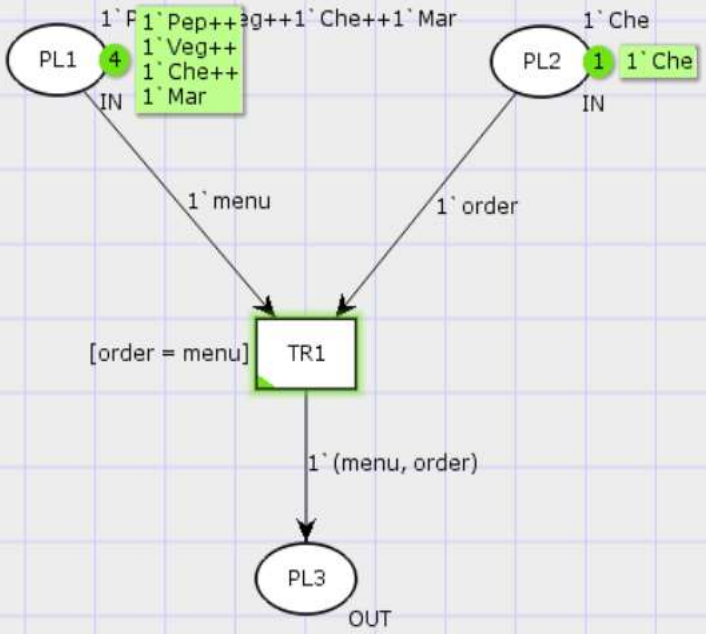


CPN Tools (Version 4.0.1, February 2015)

- Tool box
  - Auxiliary
  - Create
  - Declare
  - Hierarchy
  - Monitoring
  - Net
  - Simulation
  - State space
  - Style
  - View
- Help
- Options
- Pizza.cpn
  - Step: 0
  - Time: 0
  - Options
  - History
  - Declarations
    - Standard priorities
    - Standard declarations
      - colset UNIT
      - colset BOOL
      - colset INT
      - colset INTINF
      - colset TIME
      - colset REAL
      - colset STRING = string;
      - colset IN = with Pep | Veg | Che | Mar;
      - var menu, order : IN;
      - colset OUT
  - Monitors
  - Pizza Order

Net | Create | Style | Sim

Pizza Order — Binder 0



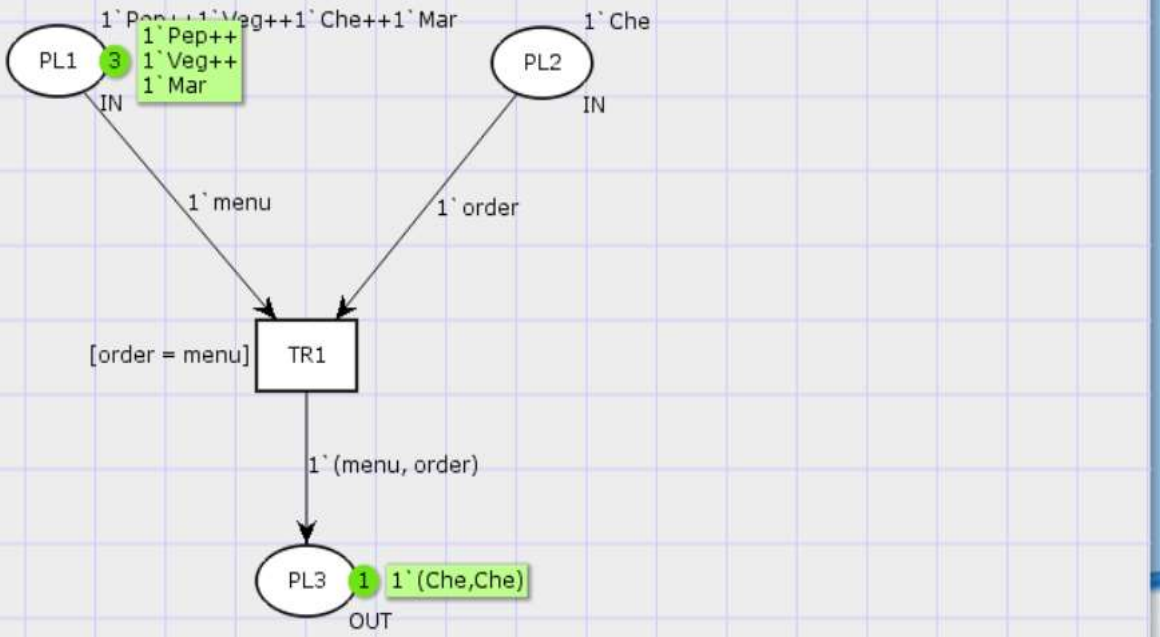
None

CPN Tools (Version 4.0.1, February 2015)

- Tool box
  - Auxiliary
  - Create
  - Declare
  - Hierarchy
  - Monitoring
  - Net
  - Simulation
  - State space
  - Style
  - View
- Help
- Options
- Pizza.cpn
  - Step: 1
  - Time: 0
  - Options
  - History
  - Declarations
    - Standard priorities
    - Standard declarations
      - colset UNIT
      - colset BOOL
      - colset INT
      - colset INTINF
      - colset TIME
      - colset REAL
      - colset STRING = string;
      - colset IN = with Pep | Veg | Che | Mar;
      - var menu, order : IN;
      - colset OUT
  - Monitors
  - Pizza Order

Net | Create | Style | Sim

Pizza Order — Binder 0

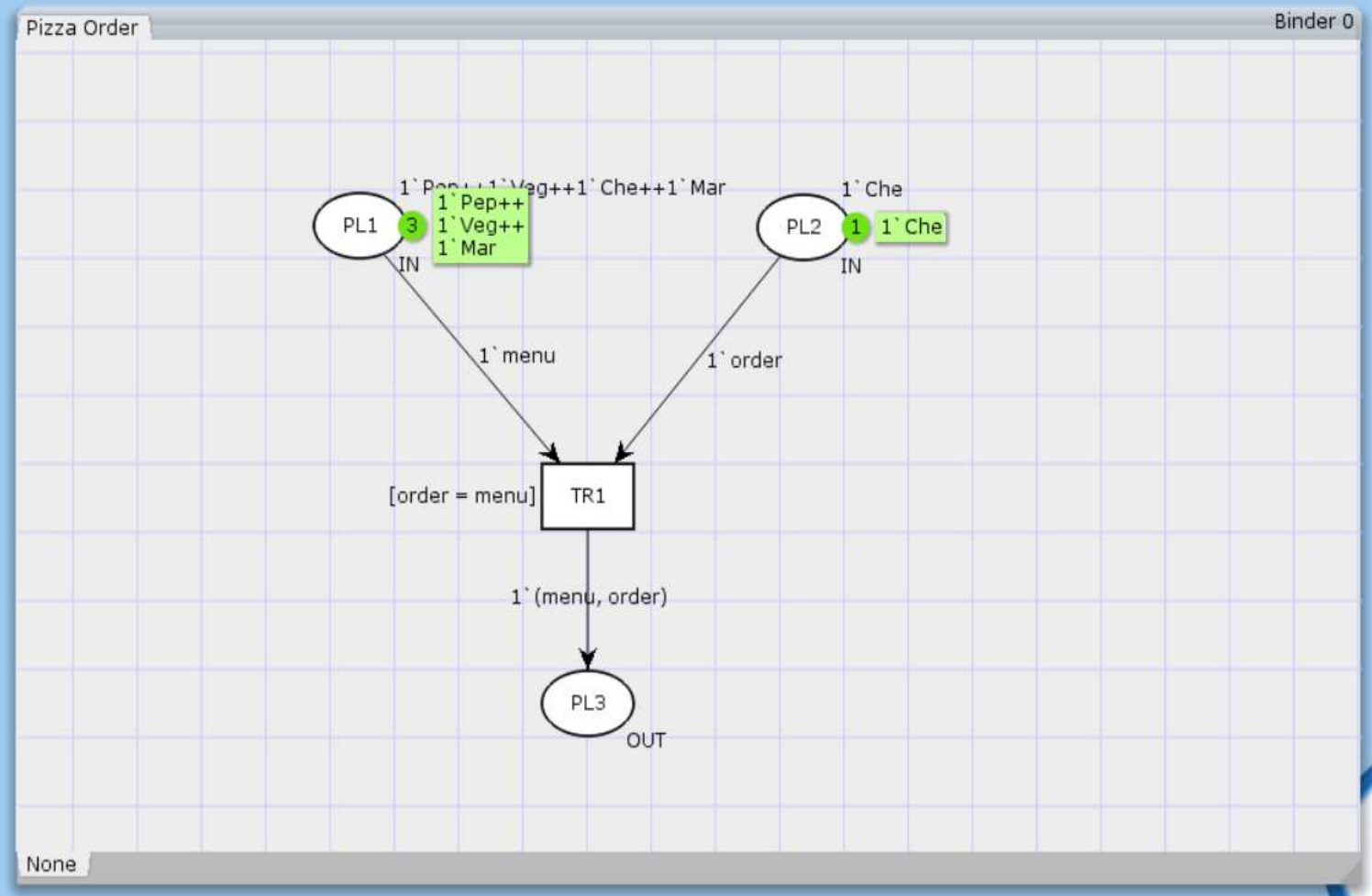


None

CPN Tools (Version 4.0.1, February 2015)

- Tool box
  - Auxiliary
  - Create
  - Declare
  - Hierarchy
  - Monitoring
  - Net
  - Simulation
  - State space
  - Style
  - View
- Help
- Options
- Pizza.cpn
  - Step: 0
  - Time: 0
  - Options
  - History
  - Declarations
    - Standard priorities
    - Standard declarations
      - colset UNIT
      - colset BOOL
      - colset INT
      - colset INTINF
      - colset TIME
      - colset REAL
      - colset STRING = string;
      - colset IN = with Pep | Veg | Che | Mar;
      - var menu, order : IN;
      - colset OUT
  - Monitors
  - Pizza Order

Net | Create | Style | Sim



Key Takeaways and Limitations

Key Takeaways

- Introduces a formal method to transform flowcharts into Colored Petri Nets(CPN)
- Defines mapping rules for Start, Process, Decision, Connector, and End nodes
- Enables simulation and validation of flowchart logic using CPN Tools

Limitations and Opportunities

- Supports only basic data types (integer, boolean, string)
- Future work could extend the model to support:
- Advanced flowchart patterns
- Tool integration for automated transformation

Conclusion by the authors

Proposed a formal scheme for transforming flowcharts into Colored Petri Nets (CPNs)

Defined mapping rules for each flowchart element: Start, Process, Decision, Connector, End

Used color sets to model basic data types (INT, BOOL, STRING)

Transformation results in a CPN model that is verifiable and executable

Validated approach using a case study to simulate and analyze flowchart behavior with CPN Tools

My comments on the paper

Clear explanation of how informal visual models can be formalized using CPNs

Mapping rules are easy to follow and logically structured

The case study demonstrates practical application and confirms the method works

Motivated me to try applying this transformation to a new flowchart of my own

Introduced me to CPN Tools and how to simulate control and data flow

Thank you!