Towards Location-Aware Process Modeling and Execution

DAB 2014, Eindhoven

- Xinwei Zhu, Wuhan University (@zhuxinwei)
- Guobin Zhu, Wuhan University
- Seppe vanden Broucke, KU Leuven (@macuyiko) (presenter)
- Jan Vanthienen, KU Leuven
- Bart Baesens, KU Leuven

xinwei.zhu@whu.edu.cn

Outline

- Introduction
- Preliminaries
- Location-Aware Workflow Modeling
- Execution of Location-Aware Models
- Prototype Implementation
- Conclusions

Outline

- **Introduction**
- Preliminaries
- Location-Aware Workflow Modeling
- Execution of Location-Aware Models
- Prototype Implementation
- Conclusions

Introduction

- Workshop theme: “taking perspectives other than control-flow into account in all stages of the BPM life cycle”

- We focus here on the perspective of “location”:
  - Features
  - Geospatial data
  - Geographic data
  - Integration with Geographic Information Systems

- Modeling and execution driven, governed by location-based data and constraints

Introduction (2)

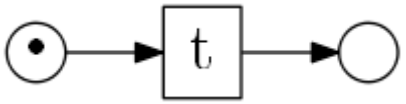
- Our methodology encompassed the following contributions:
  - Definition of WorkFlow (Petri) net model extension with location-based constraints (LAWF-net)
  - Formalization of mapping from a LAWF-net to a Coloured Petri net (CPN), for execution and validation support
- Case study developed to illustrate the validity of the approach

Outline

- Introduction
- **Preliminaries**
- Location-Aware Workflow Modeling
- Execution of Location-Aware Models
- Prototype Implementation
- Conclusions

Preliminaries

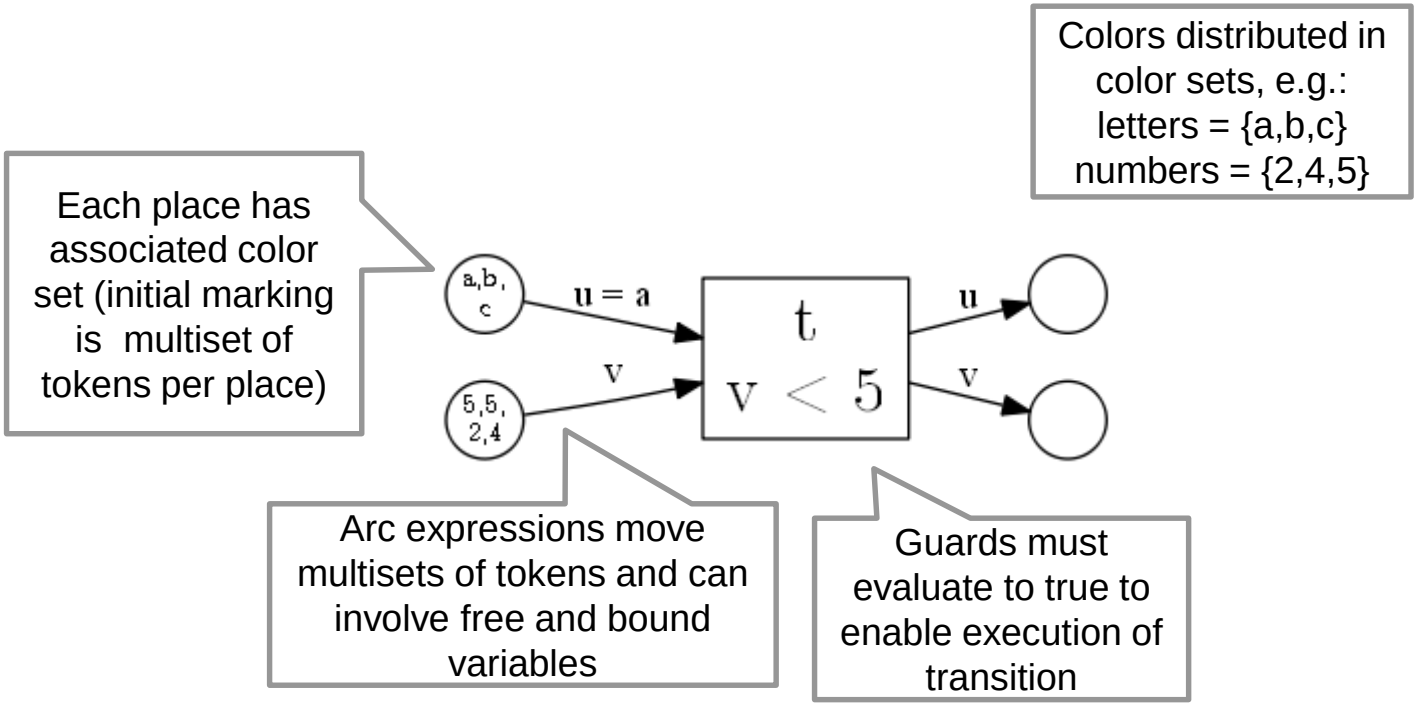
- Concept of Petri net assumed to be known:
  - Places, transitions, arcs, tokens, execution semantics (enablement, firing and movement of tokens)



- WorkFlow net (WF-net) is an extension of Petri nets:
  - Petri net is a WF-net if the model contains a single source place, sink place, and every node lies on a path from the source to sink place

Preliminaries (2)

- Coloured Petri nets are an extension of Petri nets:
  - Support for tokens of multiple types (colors)
  - Arc expressions
  - Guard conditions



Preliminaries (3)

- Features capture geospatial aspects:
  - We adhere to a broad, general definition
  - Feature belongs to certain feature type (e.g. country, city, etc.)
  - Feature needs to have attribute defining the geometry (point, line, multiline, polygon)
  - Geometry can be dynamic (e.g. moving objects)
  - Feature can contain arbitrary number of other attributes (even other features)

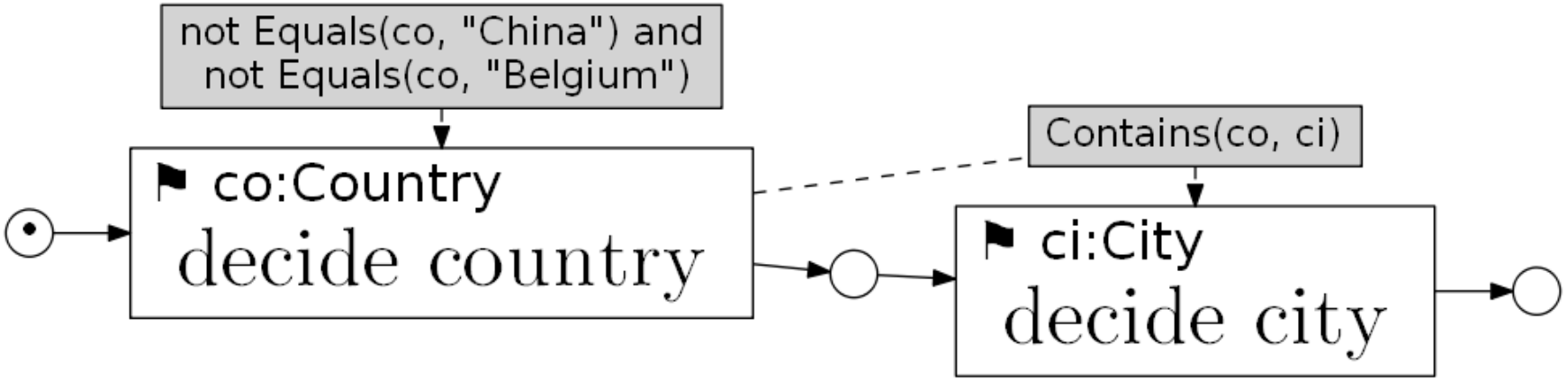
- Geospatial relationships define relations between features:
  - OGC Topic 8 and other standards
  - Feature 1 *contained in* feature 2, *equal to* feature 2, *touches* feature 2, *intersects* feature 2, etc.

Outline

- Introduction
- Preliminaries
- **Location-Aware Workflow Modeling**
- Execution of Location-Aware Models
- Prototype Implementation
- Conclusions

LAWF-net

- Extension of WF-nets
- Addition of two new constructs:
  - Location-dependent transitions (flagged transition)
  - Location constraints (shaded boxes)
  - Constraints connected with constraint flows from 0 or more location-dependent input transitions to 1 output transition



LAWF-net (2)

- Upon execution, a location is “bound” to a location-dependent transition
- Bound locations can be used as input for other constraints
- Only one location can be bound at any time; recurring transitions override previous bound locations
- Constraints act as guards for execution, if a location can be found to bind to the transition which satisfies the constraints, the transition can be executed
- Output transition can also be a non-location-dependent transition, in which case the constraints act as a constraint without selecting a location to be bound

Outline

- Introduction
- Preliminaries
- Location-Aware Workflow Modeling
- **Execution of Location-Aware Models**
- Prototype Implementation
- Conclusions

LAWF-net Execution

- Possible to define execution semantics for LAWF-nets

- But mapping to CPN formalized:
  - Validation
  - Integration with other perspectives
  - Existing tool support
  - Integration with other systems
  - More modeling flexibility (if desired)

LAWF-net to CPN Mapping

- Mapping involves the following steps:

Control-flow constructs	→	Modeled as-is (places and transitions) One control-flow token color: "*unit*"	
Color sets	→	{*U*,*FL*}	with *U*={*unit*} and *FL* the set of all locations

For each location dependent transition:

→ A "location output place" is created with type FL

→ A "location input place" is created with type FL

And with initial marking set to all location tokens belonging to that type

LAWF-net to CPN Mapping (2)

- Mapping involves the following steps:

Arcs are created as follows:

→ Arcs from and to input places to their location-dependent transitions, according to type.
The returning arc is to return the location token to the input pool.
Arc inscription = "*vl*" (input arc and returning arc)

→ Arcs from location-dependent transitions to their location output places.
This arc "binds" the location to the output place.
Arc inscription = "*vl*" (output arc)

→ Arcs from location output places to their location-dependent transitions.
This arc consumes any previously set tokens from the location output place.
(overriding arc)

→ Arcs from and to location output places to (other) location-dependent transitions.
For each location used as input in constraint in location-dependent transition.
The returning arc is to return the location token to the output place.

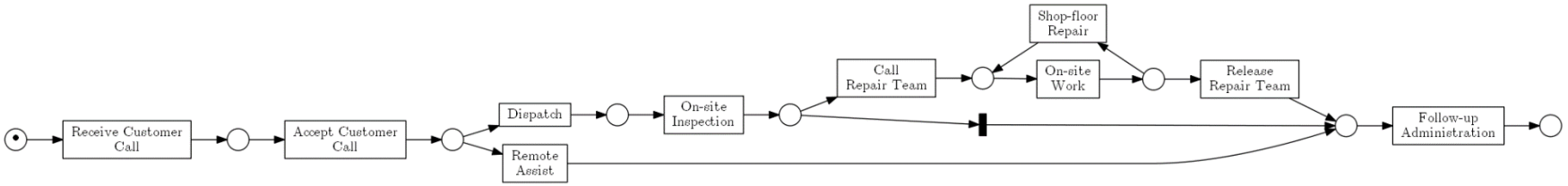
Guards → For location-dependent transitions in accordance with the LAWF-net

Outline

- Introduction
- Preliminaries
- Location-Aware Workflow Modeling
- Execution of Location-Aware Models
- **Prototype Implementation**
- Conclusions

Example Case

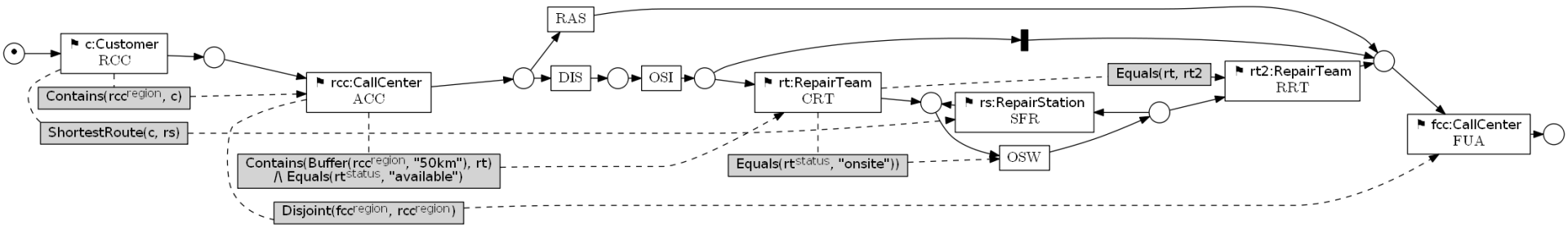
- Technical maintenance service
- Petri net without location-awareness:



- How to:
  - Ensure call centers handle customer calls within their region?
  - Call center performing follow-up work should be situated in different region than originating call center?
  - Send out repair teams and track their route?
  - Make shop-floor repairs in closest repair station?

Example Case (2)

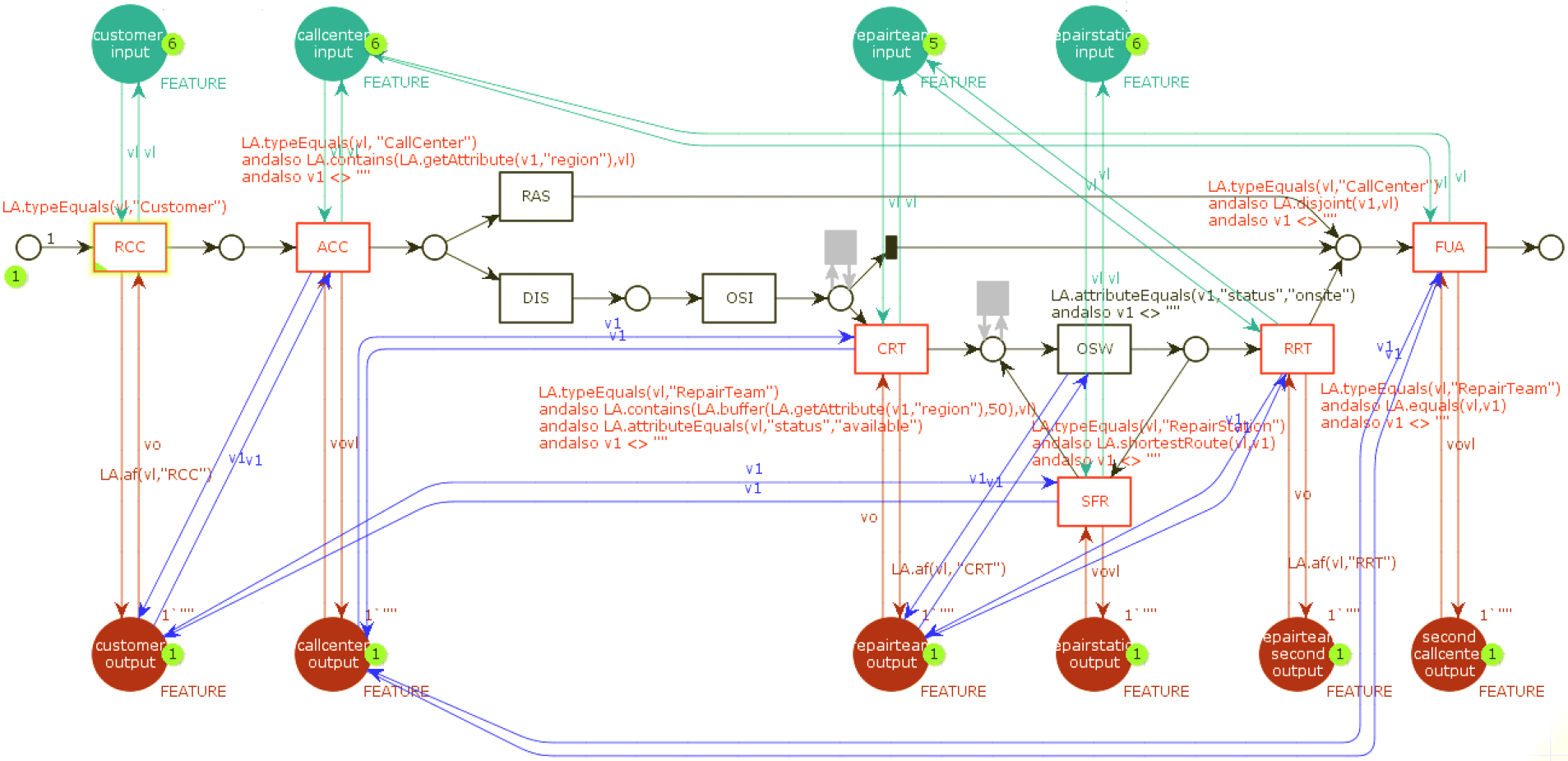
- Converted to LAWF-net:



- Accepting call center region must contain customer
- Repair station used must lie on shortest route from customer
- Repair teams can be requested only in the call center region and 50km around
- Status of repair teams must be set to “available”
- Call center handling follow up work cannot be in same region as originating call center
- On-site work can start once the repair team is “onsite”
- After the work, the repair team which performed the repair is released

Example Case (3)

- Converted to CPN:

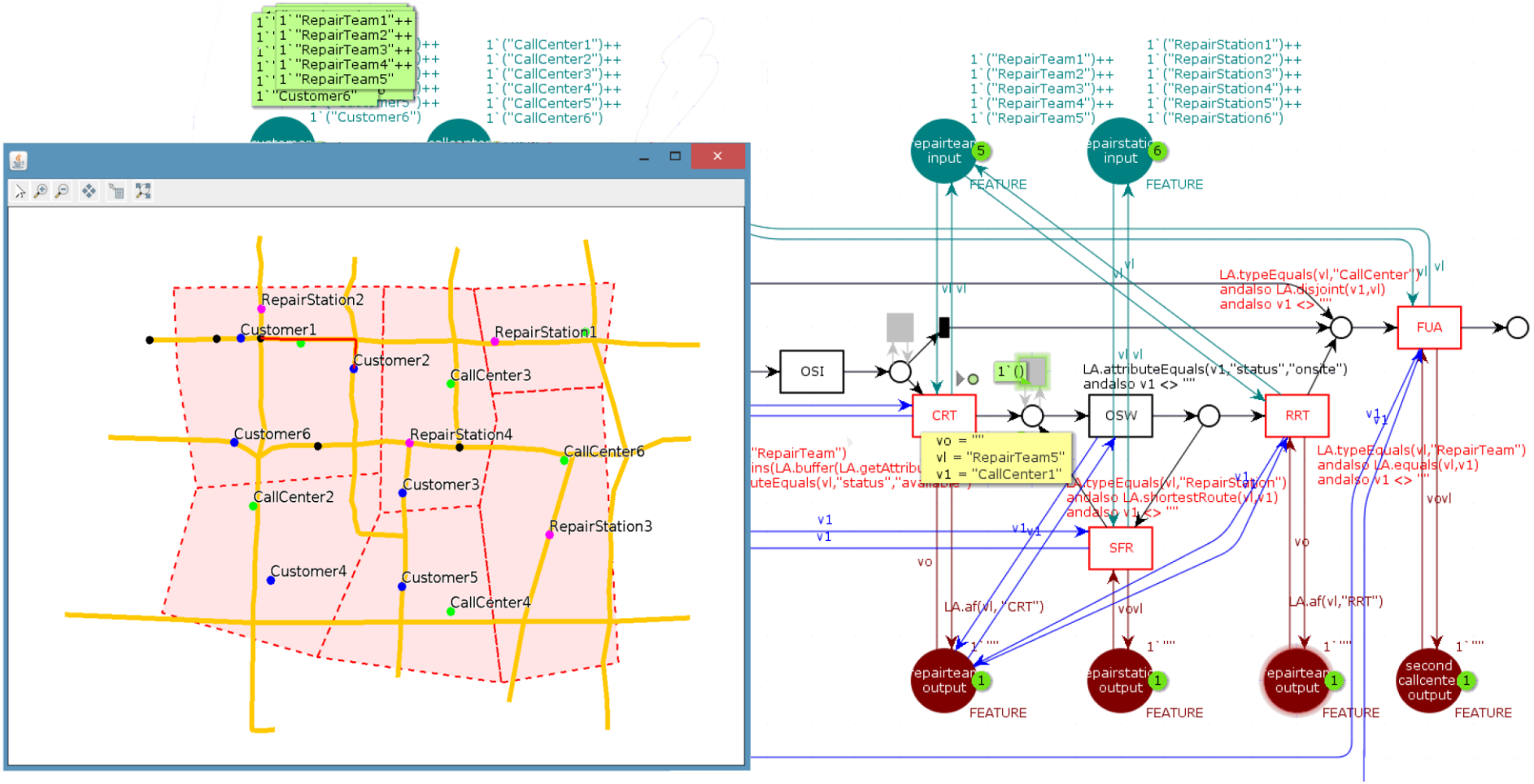


Example Case (4)

- Converted to CPN
- Constraints implemented using RPC (Remote Procedure Call) host
- Host written in Java, using GeoTools geospatial library
- Allows for easy:
  - Monitoring of running cases
  - Integration with existing GIS systems
  - Extending of constraints

Example Case (5)

- Integration with existing GIS systems feasible



Outline

- Introduction
- Preliminaries
- Location-Aware Workflow Modeling
- Execution of Location-Aware Models
- Prototype Implementation
- **Conclusions**

Conclusions

Summary:

- We have presented a WF-net extension to make modeling of location-aware processes more explicit
- As well as a means to execute such processes by means of a mapping to CPN
- Validity shown by initial case example

Extensions:

- CPN executing multiple instances at once (already implemented in example case)
- Merging of location input places (allow for more than one location type to be bound to a location-dependent transition)
- Different handling of recurrent transitions (i.e. not override but keep list)
- Allow binding more than one location to location-dependent type

Future work:

- Extend set of constraints
- Extend case study (i.e. weather data, etc.)
- Look for possibilities to combine approach with other perspectives
- Possibility to extend CPN formalization to general data-aware modeling approach: data types, values, and constraints

Questions and Answers

Thank you for your attention

Q&A