MaramaEML: An Integrated Multi-View Business Process Modelling Environment with Tree-Overlays, Zoomable Interfaces and Code Generation

Lei Li[1], John Hosking[1] and John Grundy[1,2]
[1]Department of Computer Science and [2]Department of Electrical and Computer Engineering
University of Auckland, New Zealand
{l.li,john,john-g }@cs.auckland.ac.nz

Abstract

MaramaEML is an integrated support tool for the Enterprise Modelling Language (EML) built using the Eclipse based Marama framework. It provides support for multiple visual notations including: the Business Process Modelling Notation (BPMN); the EML tree-based, multi-layer hierarchical service representation; fisheye zooming capabilities; automatic BPEL code generation; and inter-notation mapping.

1. Enterprise Modelling Language

Most current approaches to modelling complex business processes fail to scale for large processes and organizations. Key issues include: (1) cobweb and labyrinth problems exhibited by conventional box and line metaphors and large numbers of hidden dependencies introduced by compartment-based modularity (e.g. BPMN, FormChart, BioOpera and ZenFlow [1][3][8][9]); (2) inefficiencies in presenting multi-level abstractions of business processes (e.g. WTD and T-Web systems [6]); (3) notations too complex for a business user to learn and use (e.g. UML [2]); and (4) requiring substantial programming knowledge (e.g. ARIS and TOVE [4]).

We have been developing a new approach, the Enterprise Modelling Language, based on trees, overlays and fish-eye viewers to overcome the shortcomings of existing workflow notations. EML is the first tree overlay structure visual language in the area of business process modeling. Service architectures are represented as trees and business sequences are modeled as process overlays on the service trees. By combining these two mechanisms EML gives users a clear overview of an enterprise system structure while business processes are modeled by overlays on the same view. EML uses a multi layer structure to model business processes, exception handlers and dependency triggers on different levels. This approach significantly reduces the complexity of business processes. Please refer to [7] for a detailed description of EML.

2. MaramaEML

We have developed an integrated design environment, MaramaEML for creating EML specifications. This IDE provides a platform for: efficient EML visual model creation, inspection, editing, and storage, model driven code generation, and integration with other diagram types. Figure 1 shows an overview of whole MaramaEML environment structure. MaramaEML's major features include:

(1) MaramaEML permits efficient production of EML visual models (a) facilitating their creation, display, editing, storage, code generation and integration with other diagrams;

(2) Due to the complexity of business processes, a single modelling notation is insufficient to satisfy all modelling needs. We believe that the ideal solution is to provide the user access to an integrated visual modelling environment. The MaramaEML support tool includes concurrent EML (Figure1 (a)), BPMN ((b) and Form Chart (c) views, with BPEL scripts (d).

(3) MaramaEML supports automatic BPEL code (d) generation from BPMN, EML and Form-Chart views; this involves model dependency analysis and its translation to structured activity constructs. The form chart view can also be compiled to client-side JavaScript which defines page navigations and client-server communications.

(4) The multiple modeling notations are integrated together using generated XML-based BPEL scripts as the interchange format. This provides multi-level support for integrating complex enterprise system models.

(5) We have integrated a 3rd party LTSA engine to verify the correctness of generated BPEL code. This engine compiles EML output (BPEL) displaying results in a MaramaEML panel. If there are no compilation errors, an LTS diagram (Labelled Transition System) is shown (e).

(6) A distortion-based fisheye zooming function has been implemented in MaramaEML to enhance complex diagram scalability and navigability.

MaramaEML is implemented using our Marama meta-tool as a set of Eclipse plug-ins, providing a robust and scalable design tool. We specified the EML, BPMN and Form Chart domain-specific visual language notations and meta-models and generated Eclipse-based editors from these to realise the basic support environment. The tree layout, overlays and distortion-based displays are all implemented as complex visual event handlers. The integration of EML with BPMN notation, code generation of BPEL, and LSTA engine integration are implemented as event-driven, model-level data updates.

Performance simulation [5] is also incorporated in the integrated MaramaEML, facilitating cost-effective tests of the integrated specifications using random data with visualisation of test results using the same design-level specification views.

3. Evaluation

We have conducted three evaluations for MaramaEML. The first was an extensive cognitive dimensions analysis guiding the design and implementation. It was undertaken by the MaramaEML designer with closeness of mapping and hidden dependency mitigation emphasised. The second was a small scale task-based end-user evaluation of an early release version. The objective was to assess how easy it was to learn to use MaramaEML and how efficiently it can solve the diagram complexity problem as input to refining the software. The third evaluation was a large formal end user evaluation of the most recent release. Feedback suggests MaramaEML is very straightforward to use and understand. Users feel the tree overlay method greatly reduces the complexity of modelling business processes compared to using only conventional BPMN views. The automatic code generation and multi-view collaboration were seen as enhancing the modelling strength. The zooming function is quite easy to use and increases the navigation ability evidently.

4. References

[1] BPMI, http://www.ebpml.org/bpml.htm, 2006

[2] C. Marshall, *Enterprise Modelling with UML. Designing Successful Software Through Business Analysis,* Addison Wesley, 2000

[3] D. Draheim and G. Weber, *Form-Oriented Analysis*, Springer-Verlag Berlin Heidelberg, 2005

[4] F. Buschmann, R. Meunier, and H. Rohnert, etc, *Pattern-Oriented Software Architecture*. John Wiley and Sons, 1996

[5] J.C., Grundy, J.G.,Hosking, L., Li and N. Liu, "Performance Engineering of Service Compositions", *Proceeding of ICSE2006*, p26 ~ p32, Shanghai, China

[6] J. Kornkamol, S. Tetsuya, and T. Takehiro, "A Visual Approach to Development of Web Services Providers/Requestors", *Proc. VL/HCC'03* p251~p253

[7] L. Li, J.G. Hosking and J.C. Grundy, EML, A Tree Overlay-based Visual Language for Business Process Modeling, *Proceeding of ICEIS 07*, 13~19, Funchal, Madeira, Portugal

[8] Martinez, A.; Patino-Martinez, M.; Jimenez-Peris, R.; Perez-Sorrosal, F., ZenFlow: a visual Web service composition tool for BPEL4WS, Proc IEEE VLHCC'05 181-188, 2005

[9] Pautasso, C.; Alonso, G., Visual composition of Web services, Proc IEEE HCC'03. 92-99,. 2003

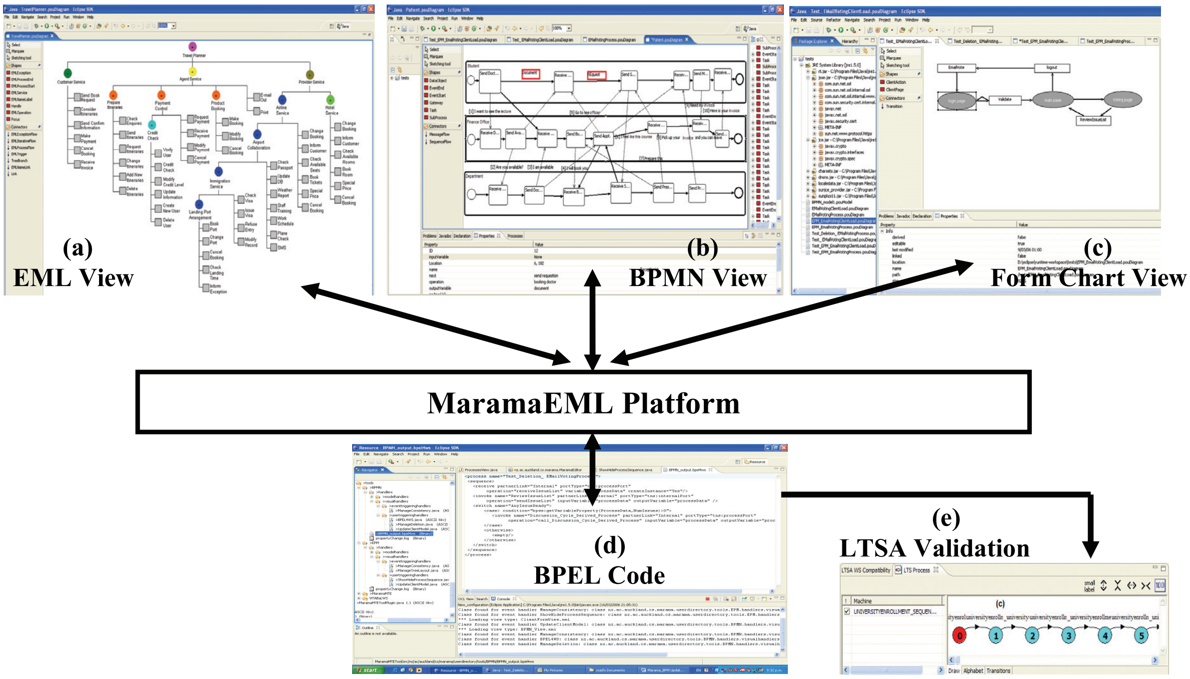


Figure 1. MaramaEML Integrated Environment

Appendix:
Brief Presentation Description

Part 1: Brief EML Introduction

At the start of the presentation, the major concepts and semantic meaning of the Enterprise Modelling Language will be introduced via a set of examples. This will include a short motivation, modelling elements, the tree-overlay structure, process flows, exception handling, service reuse, dependency/internal exceptions, iteration and selection.

Part 2: Detailed MaramaEML Demo

A detailed software demonstration will be introduced in this part. This will focus on a process modeling scenario with MaramaEML. Key tasks will include:

- extend an existing EML tree; (A1)
- collapse/expand service nodes in an EML tree
- show a process overlay on a service tree (A2)
- show a trigger overlay on a service tree (A3)
- show an exception handling overlay (A4)
- integration of the overlays
- show a BPMN diagram in MaramaEML (A5)
- demonstrate multi-view collaboration (A6)
- demonstrate BPEL code generation (A7)
- demonstrate LTSA validation of BPEL (A7)
- demonstrate fisheye/zooming support (A8)

If time permits, some optional features will be shown

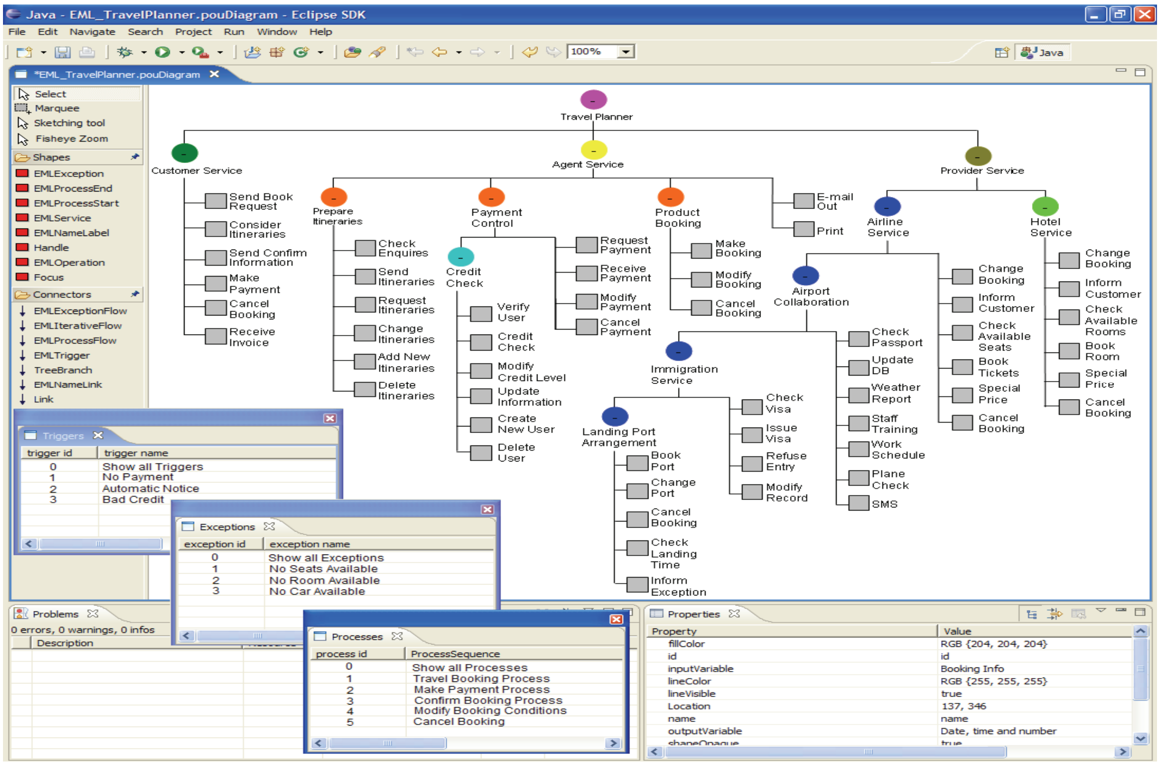
- Form chart modelling
- Form chart diagram compilation to client side javascript

Part 3: Evaluation Results and Discussion

We will conclude the presentation with evaluation results (A9) and discussion of future work.

Maturity and Availability

Marama is open source and is available for download - https://wiki.auckland.ac.nz/display/csidst. It has been used to develop a wide range of multi-view, domain-specific visual language tools. MaramaEML is not currently available but will be made available for download in the near future.



Triggers

trigger id	trigger name
0	Show all Triggers
1	No Payment
2	Automatic Notice
3	Bad Credit

Exceptions

exception id	exception name
0	Show all Exceptions
1	No Seats Available
2	No Room Available
3	No Car Available

Processes

process id	ProcessSequence
0	Show all Processes
1	Travel Booking Process
2	Make Payment Process
3	Confirm Booking Process
4	Modify Booking Conditions
5	Cancel Booking

0 errors, 0 warnings, 0 infos

Property	Value
fillColor	RGB {204, 204, 204}
id	id
inputVariable	Booking Info
lineColor	RGB {255, 255, 255}
lineVisible	true
Location	137, 346
name	name
outputVariable	Date, time and number
shapeOpaque	true

Figure A1: EML Tree in MaramaEML

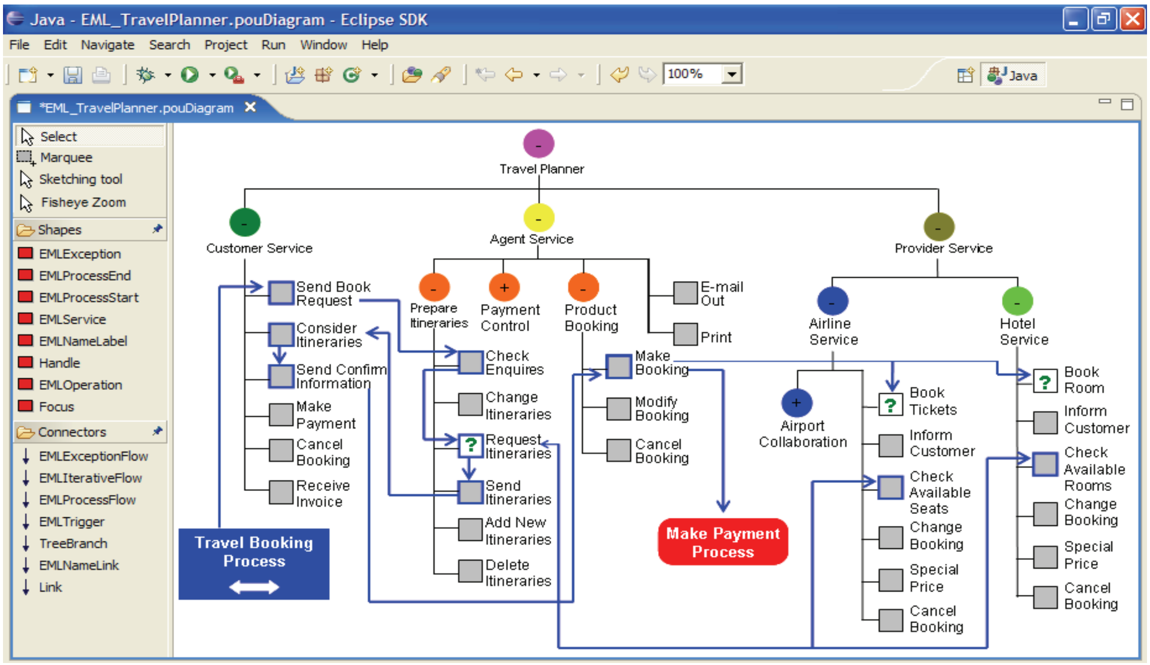


Figure A2: Process Overlay

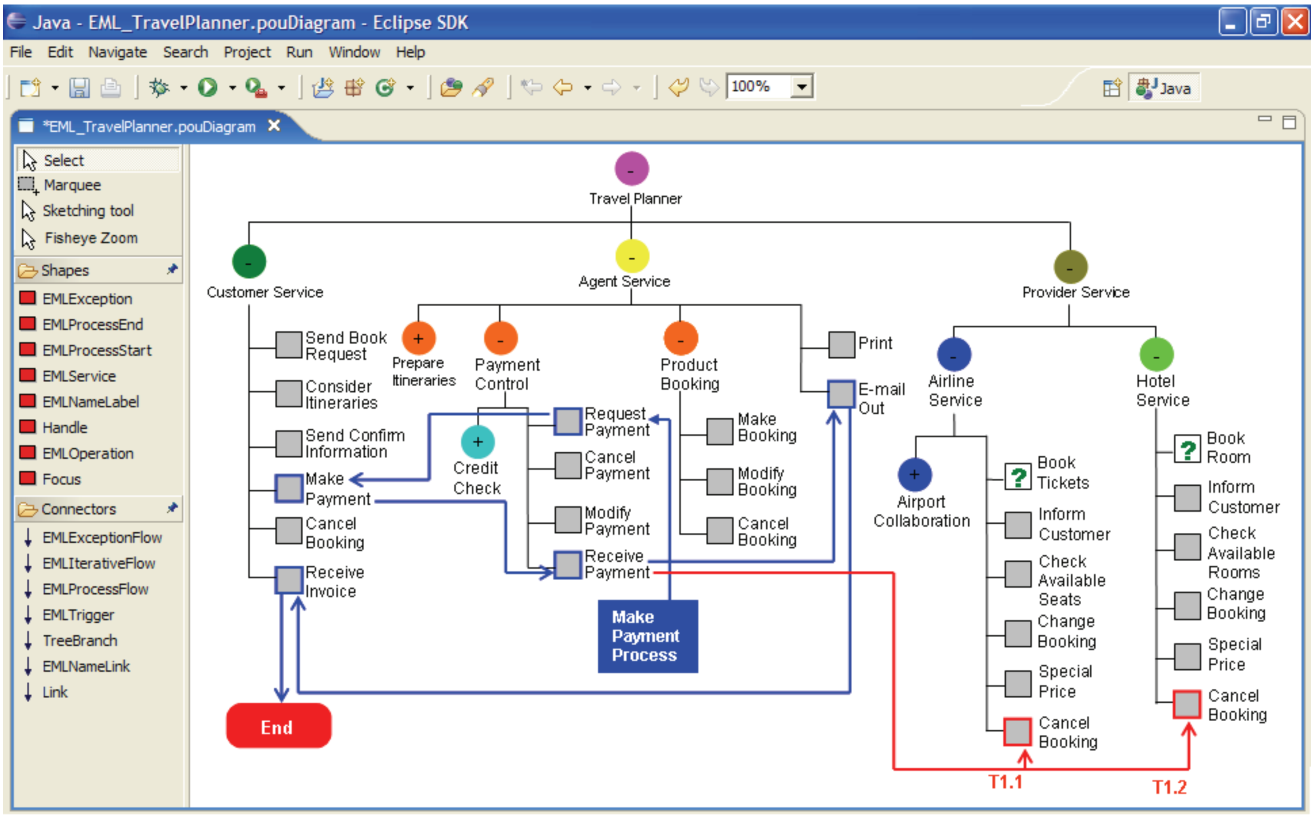


Figure A3: Triggers

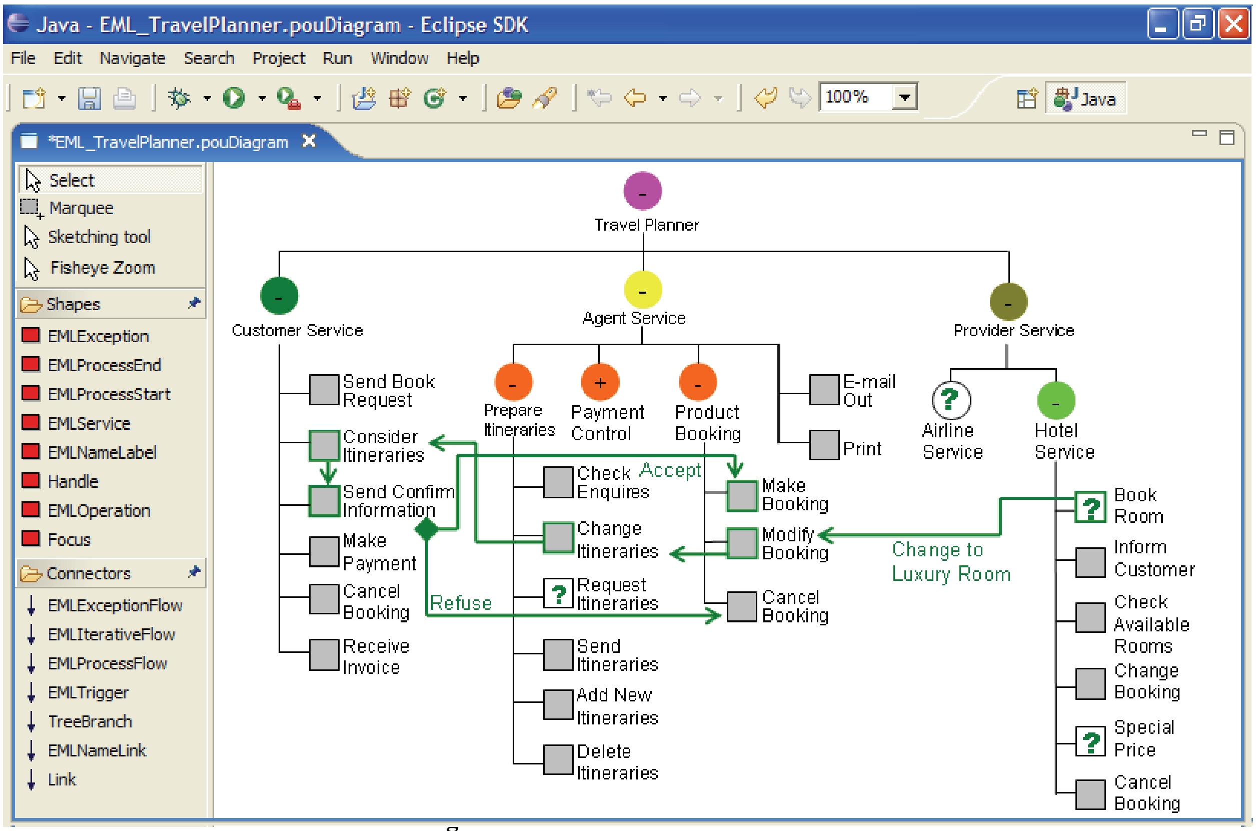
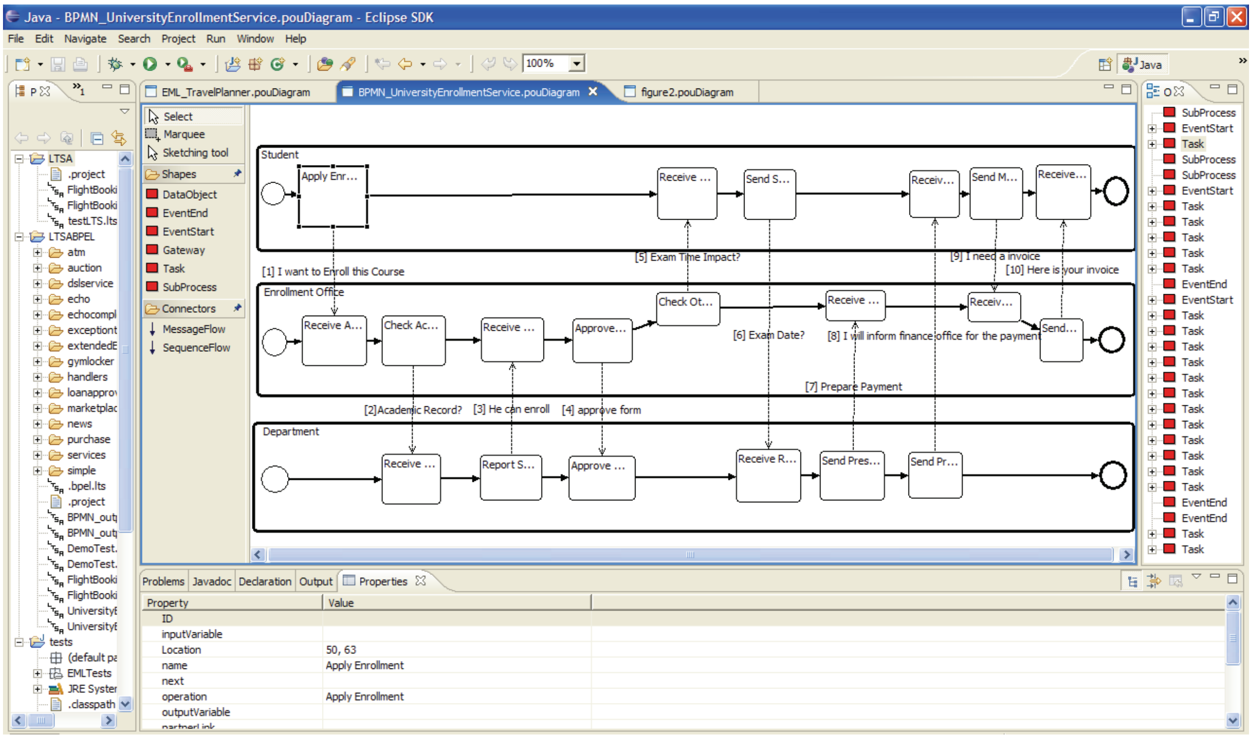
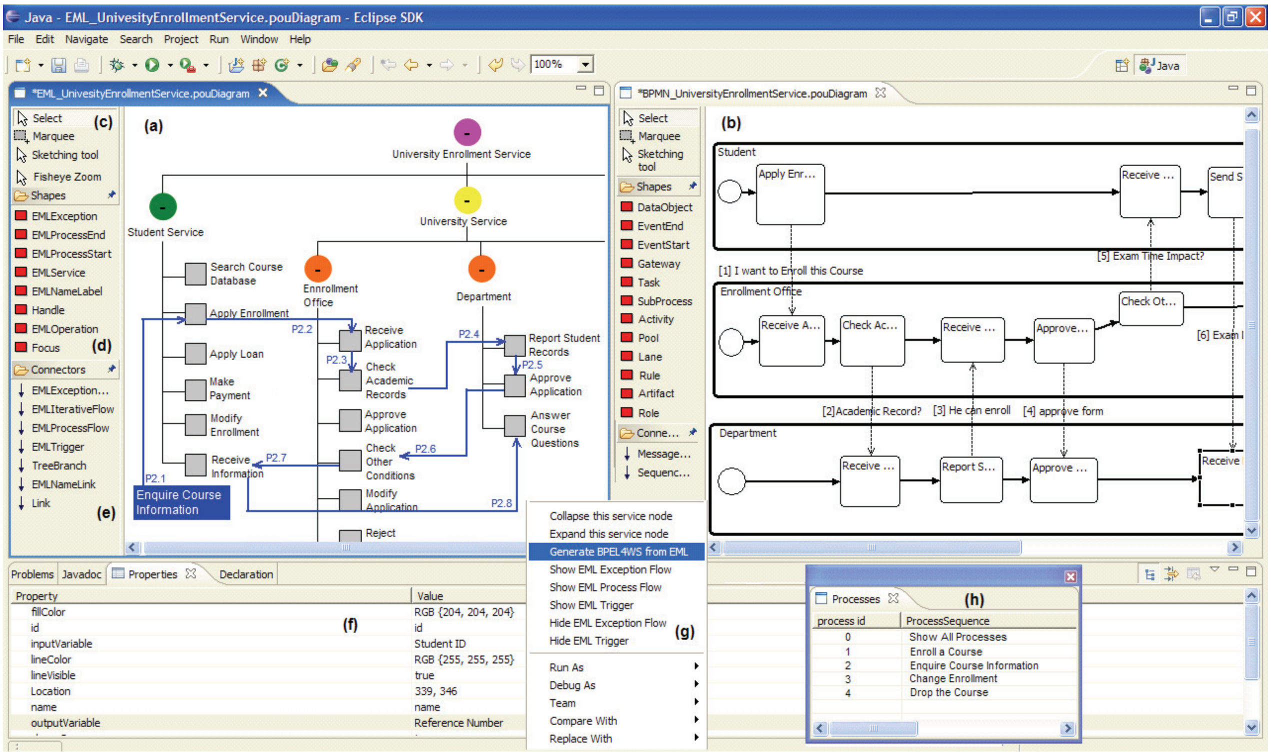


Figure A4: Exception Handling Layer



Property	Value
ID	
inputVariable	
Location	50, 63
name	Apply Enrollment
next	
operation	Apply Enrollment
outputVariable	
partnerLink	

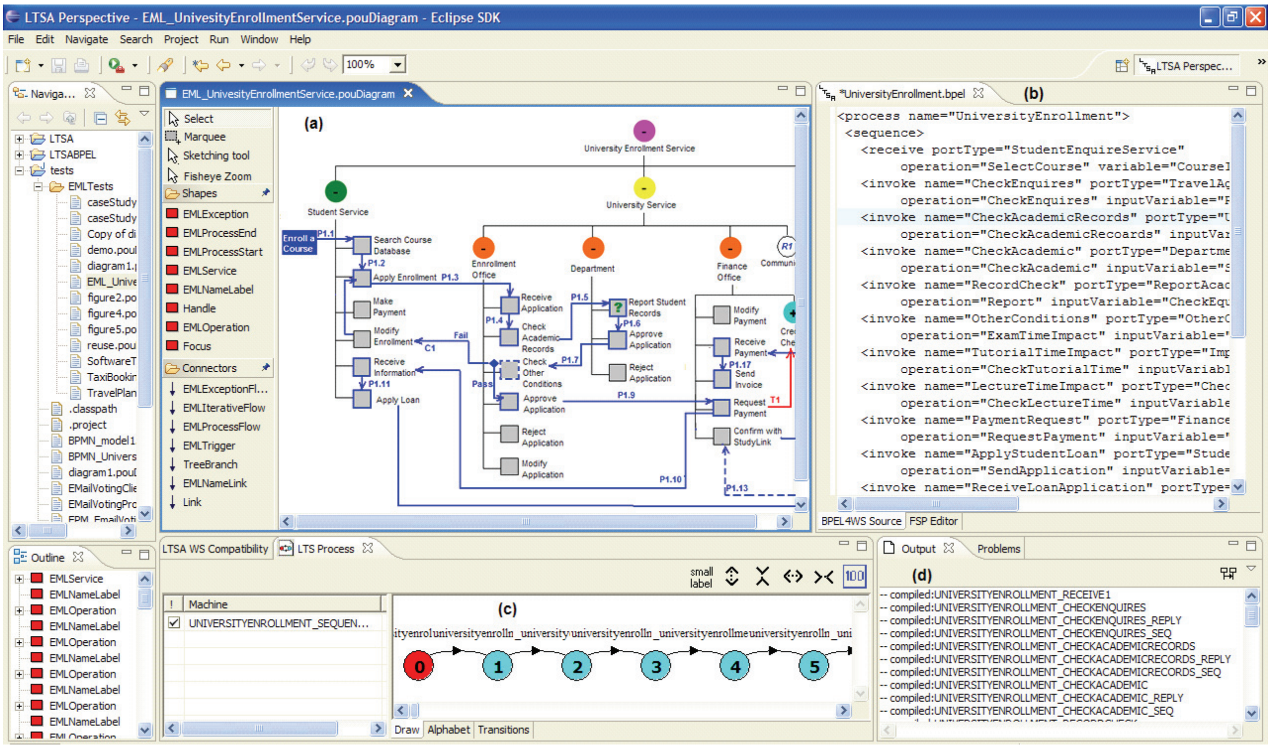
Figure A5: BPMN Diagram in MaramaEML



Property	Value
fillColor	RGB {204, 204, 204}
id	id
inputVariable	Student ID
lineColor	RGB {255, 255, 255}
lineVisible	true
Location	339, 346
name	name
outputVariable	Reference Number

process id	ProcessSequence
0	Show All Processes
1	Enroll a Course
2	Enquire Course Information
3	Change Enrollment
4	Drop the Course

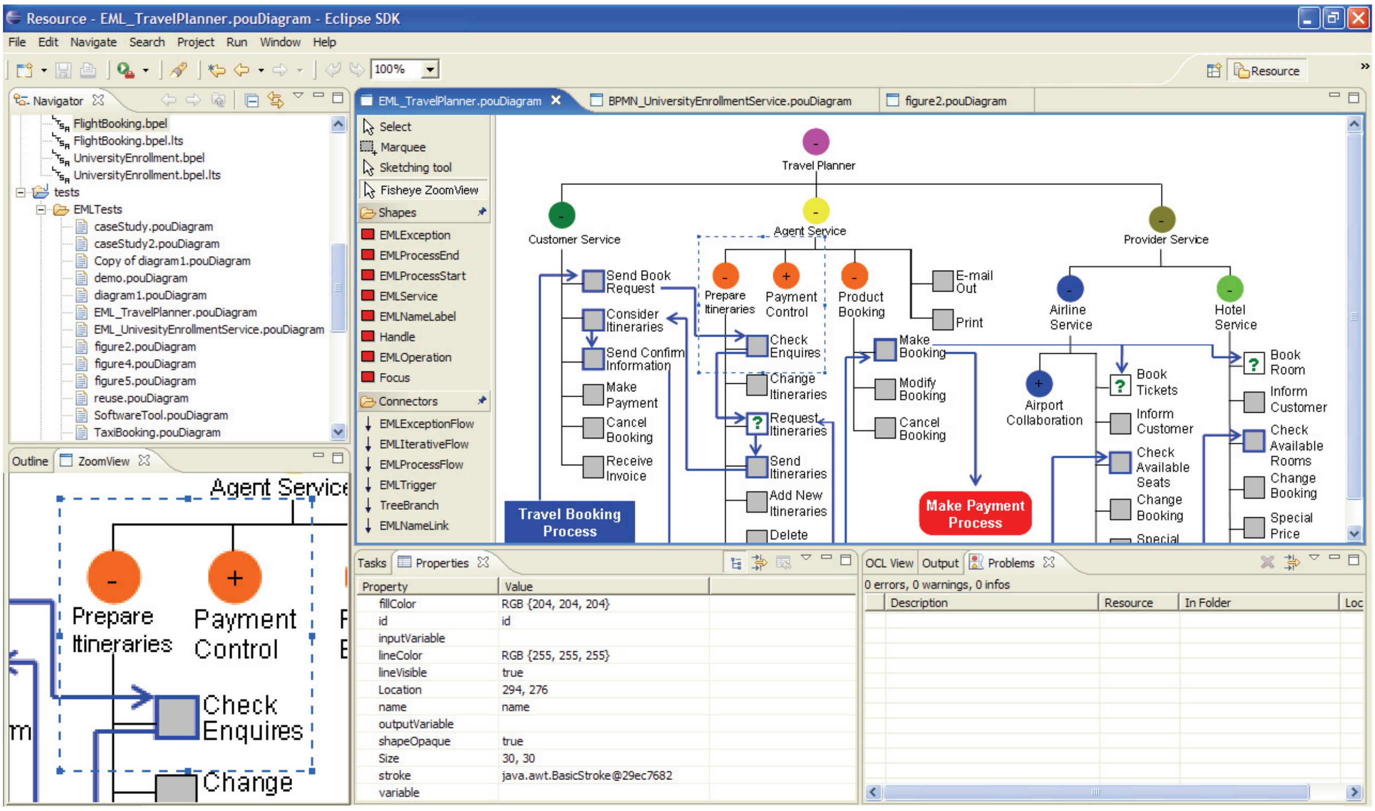
Figure A6: Multi-View Support (EML & BPMN Collaboration)



```
<process name="UniversityEnrollment">
 <sequence>
   <receive portType="StudentEnquireService"
        operation="SelectCourse" variable="CourseI
   <invoke name="CheckEnquires" portType="TravelAg
        operation="CheckEnquires" inputVariable="F
   <invoke name="CheckAcademicRecords" portType="U
        operation="CheckAcademicRecoards" inputVar
   <invoke name="CheckAcademic" portType="Departme
        operation="CheckAcademic" inputVariable="S
   <invoke name="RecordCheck" portType="ReportAcac
        operation="Report" inputVariable="CheckEqu
   <invoke name="OtherConditions" portType="OtherC
        operation="ExamTimeImpact" inputVariable="
   <invoke name="TutorialTimeImpact" portType="Imp
        operation="CheckTutorialTime" inputVariabl
   <invoke name="LectureTimeImpact" portType="Chec
        operation="CheckLectureTime" inputVariable
   <invoke name="PaymentRequest" portType="Finance
        operation="RequestPayment" inputVariable="
   <invoke name="ApplyStudentLoan" portType="Stude
        operation="SendApplication" inputVariable=
   <invoke name="ReceiveLoanApplication" portType=
```

```
-- compiled:UNIVERSITYENROLLMENT_RECEIVE1
-- compiled:UNIVERSITYENROLLMENT_CHECKENQUIRES
-- compiled:UNIVERSITYENROLLMENT_CHECKENQUIRES_REPLY
-- compiled:UNIVERSITYENROLLMENT_CHECKENQUIRES_SEQ
-- compiled:UNIVERSITYENROLLMENT_CHECKACADEMICRECORDS
-- compiled:UNIVERSITYENROLLMENT_CHECKACADEMICRECORDS_REPLY
-- compiled:UNIVERSITYENROLLMENT_CHECKACADEMICRECORDS_SEQ
-- compiled:UNIVERSITYENROLLMENT_CHECKACADEMIC
-- compiled:UNIVERSITYENROLLMENT_CHECKACADEMIC_REPLY
-- compiled:UNIVERSITYENROLLMENT_CHECKACADEMIC_SEQ
```

Figure A7: BPEL Code Generation & LTSA Validation



Property	Value
fillColor	RGB {204, 204, 204}
id	id
inputVariable	
lineColor	RGB {255, 255, 255}
lineVisible	true
Location	294, 276
name	name
outputVariable	
shapeOpaque	true
Size	30, 30
stroke	java.awt.BasicStroke@29ec7682
variable	

0 errors, 0 warnings, 0 infos

Description	Resource	In Folder	Loc

Figure A8: Zoom View Support in EML Tree

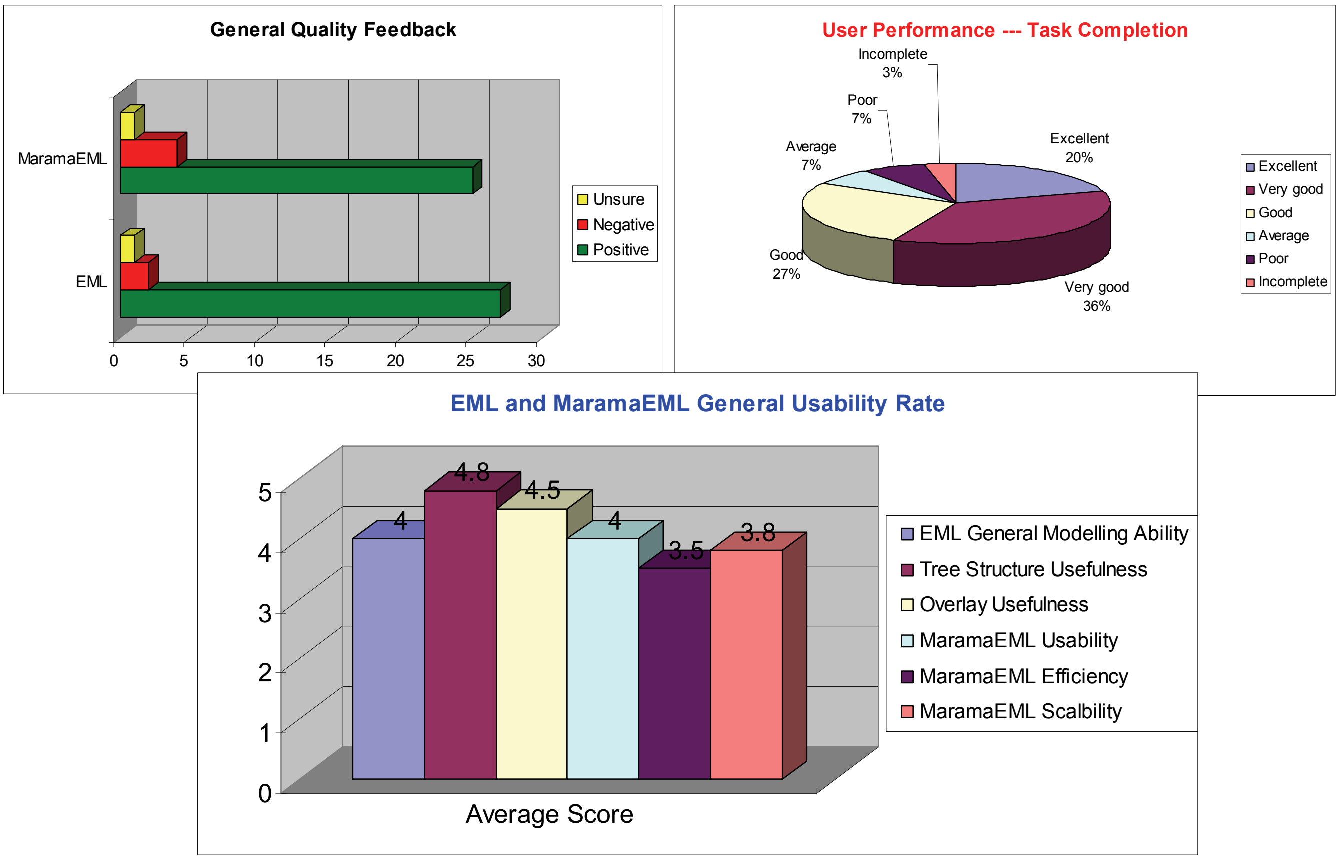


Figure A9: Formal Evaluation Feedback Summary