IWCPS 2012
Daegu, Korea

Formal Verification of ECML using HyTech
(ECML: ETRI CPS Modeling Language)

JUNBEOM YOO
KONKUK University
2012.08.23

DEPENDABLE SOFTWARE
LABORATORY

Contents

- Project Motivation

- CPS Modeling & Verification Techniques
  - ECML (ETRI CPS Modeling Language)
  - HyTech

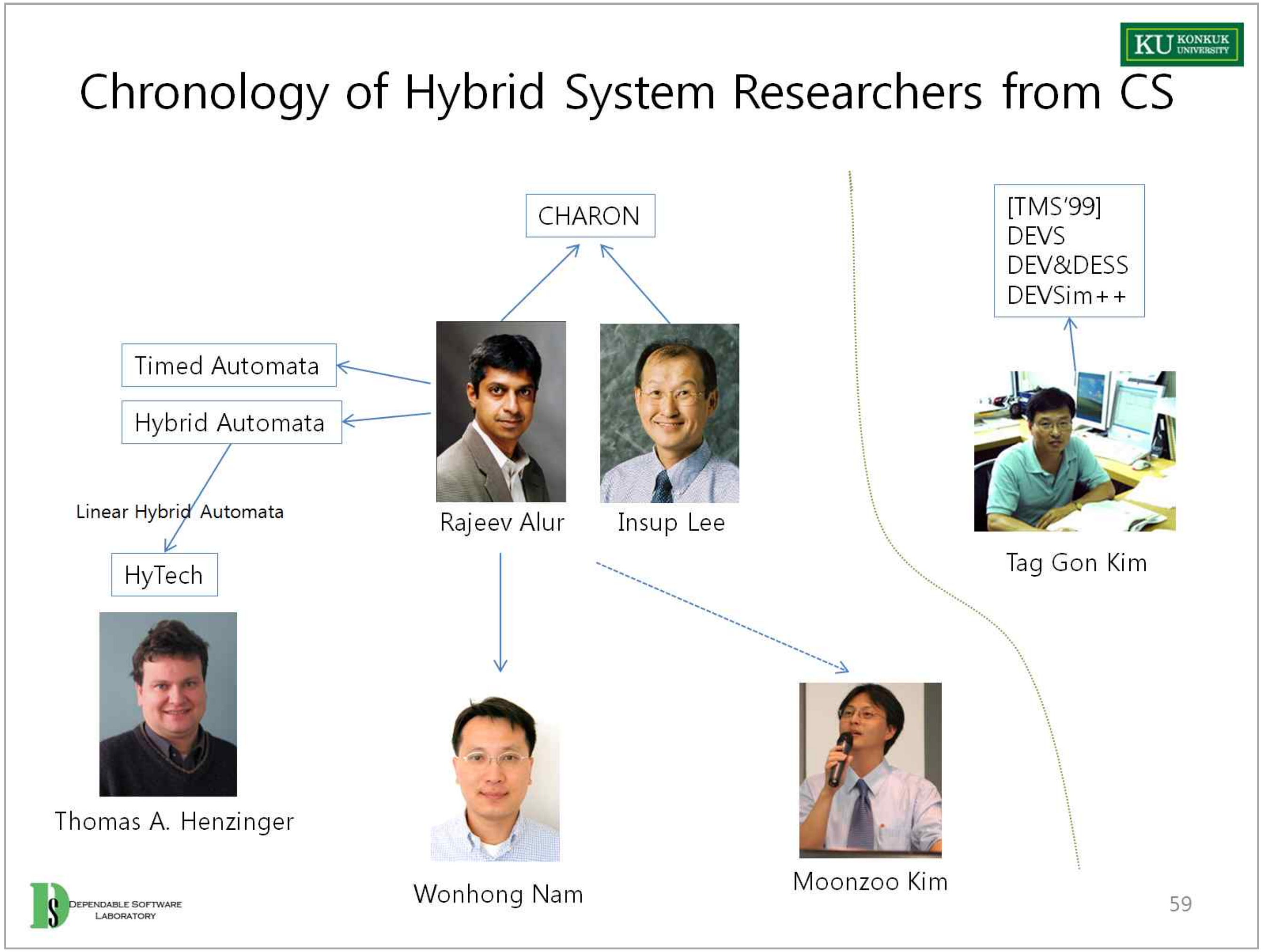
- Formal Verification of ECML using HyTech
  - ECMLtoHyTech Translator
  - HyTech Analyzer

- Conclusion and Future Work

Project Motivation

- ETRI CPS Team
  - Developing a framework of CPS modeling, simulation and verification, from 2010
  - Proposed a new CPS modeling language – ECML
  - Not yet supporting formal verification/Analysis of ECML
- KONKUK University
  - Joined the ETRI CPS Project in 2011
  - Trying to develop a way to verify ECML models with existing CPS verification tools
  - Much troubled, since we didn't know hybrid systems.

Our Effort for Finding a Research Staring Point



Chronology of Hybrid System Researchers from CS

CHARON

Timed Automata

Hybrid Automata

Linear Hybrid Automata

HyTech

Thomas A. Henzinger

Rajeev Alur

Insup Lee

Wonhong Nam

Moonzoo Kim

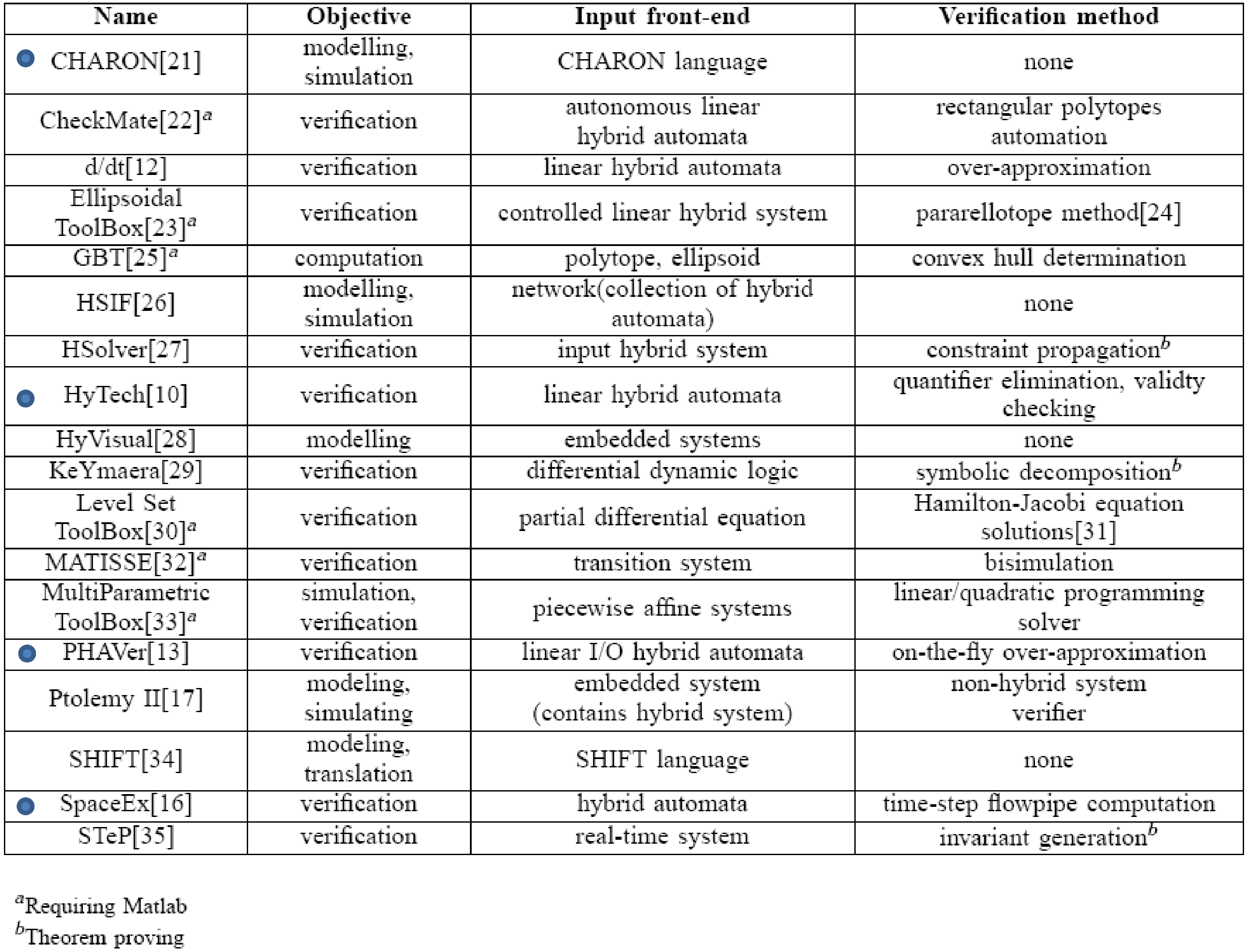
[TMS'99]
DEVS
DEV&DESS
DEVSim++

Tag Gon Kim

59

Excerpted from a presentation to ETRI in 2011.06

CPS Modeling & Verification Techniques



Name	Objective	Input front-end	Verification method
● CHARON[21]	modelling, simulation	CHARON language	none
CheckMate[22][a]	verification	autonomous linear hybrid automata	rectangular polytopes automation
d/dt[12]	verification	linear hybrid automata	over-approximation
Ellipsoidal ToolBox[23][a]	verification	controlled linear hybrid system	pararellotope method[24]
GBT[25][a]	computation	polytope, ellipsoid	convex hull determination
HSIF[26]	modelling, simulation	network(collection of hybrid automata)	none
HSolver[27]	verification	input hybrid system	constraint propagation[b]
● HyTech[10]	verification	linear hybrid automata	quantifier elimination, validty checking
HyVisual[28]	modelling	embedded systems	none
KeYmaera[29]	verification	differential dynamic logic	symbolic decomposition[b]
Level Set ToolBox[30][a]	verification	partial differential equation	Hamilton-Jacobi equation solutions[31]
MATISSE[32][a]	verification	transition system	bisimulation
MultiParametric ToolBox[33][a]	simulation, verification	piecewise affine systems	linear/quadratic programming solver
● PHAVer[13]	verification	linear I/O hybrid automata	on-the-fly over-approximation
Ptolemy II[17]	modeling, simulating	embedded system (contains hybrid system)	non-hybrid system verifier
SHIFT[34]	modeling, translation	SHIFT language	none
● SpaceEx[16]	verification	hybrid automata	time-step flowpipe computation
STeP[35]	verification	real-time system	invariant generation[b]

[a] Requiring Matlab

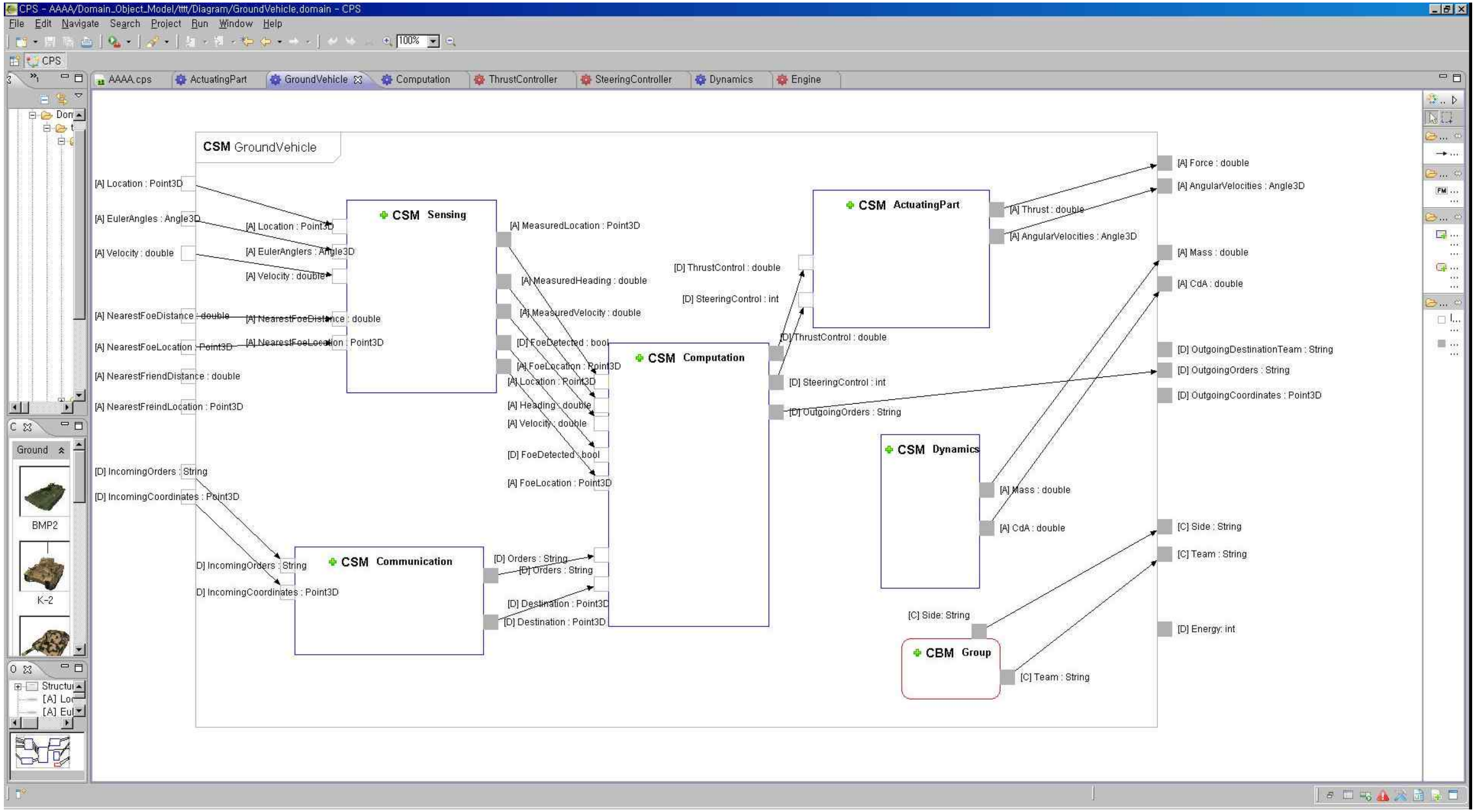
[b] Theorem proving

Name	Year (Update)	Tool Support	Execution Environment	Functions (M/S/A/V/Tr)	Verifiability	Input Front-End	Verification Technique
CHARON	2001	Yes	JAVA	M / S	N/A	Automata	N/A
CheckMate	-	No	MATLAB	V	MATLAB	MATLAB	Approximate quotient transition systems
d/dt	2001	Yes	Linux	M / S	-	d/dt input language	Forward reachability analysis
Ellipsoidal Toolbox	2006	Yes	MATLAB	V	MATLAB	MATLAB	Forward and backward reachability analysis
GBT	2004	Yes (Commercial)	MATLAB	A	MATLAB	MATLAB	Convex hull
HSIF	2002	Yes	Windows	M / S	N/A	GME model	N/A
HSolver	2005	Yes	Linux	V	Manual	Input program	Theorem provin (Rsolver)
HyTech	**2000**	**Yes**	**Linux**	**V**	**Automatic**	**Linear hybrid automata**	**Polyhedral model checking**
HyVisual	2000 (2005)	Yes	JAVA	M / S	N/A	Ptolemy plug-in	N/A
Hybrid ToolBox	2004 (2011)	Yes	MATLAB	M / S / V	MATLAB	HYSDEL language, MATLAB	LP/QP Solver
HYSDEL	2002 (2011)	Yes	Windows, Linux, Solaris	Tr	N/A	HYSDEL language	N/A
KeYmaera	2006 (2011)	Yes	JAVA	V	Manual	Differential dynamic logic formula	Theorem Proving (KeY)
Level Set Toolbox	2004 (2011)	Yes	MATLAB	S / V	MATLAB	MATLAB	Set of Algorithms
MATISSE	2005	Yes	MATLAB	V	MATLAB	MATLAB	Bi-simulation, reachable analysis
MultiParametric Toolbox	2004 (2006)	Yes	MATLAB	M / A / V	MATLAB	MATLAB	Forward and backward reachability analysis
PHAVer	**2004 (2007)**	**Yes**	**Windows, Linux, Mac**	**V**	**Automatic**	**Linear hybrid automata**	**Forward and backward reachability analysis**
Ptolemy	**2002 (2010)**	**Yes**	**JAVA**	**M / A / V**	**Automatic MATLAB**	**UML (in XML), Java code, MATLAB**	**SMV**
SHIFT	1999	Yes	Linux	M / Tr	N/A	Shift language	N/A
SpaceEx	**2010 (2011)**	**Yes**	**Linux**	**V**	**Automatic**	**SX language**	**LeGuernic-Girard Algorithm**
STeP	**1994 (1998)**	**Yes**	**Linux**	**V**	**Automatic**	**STeP language**	**Deductive model checking**

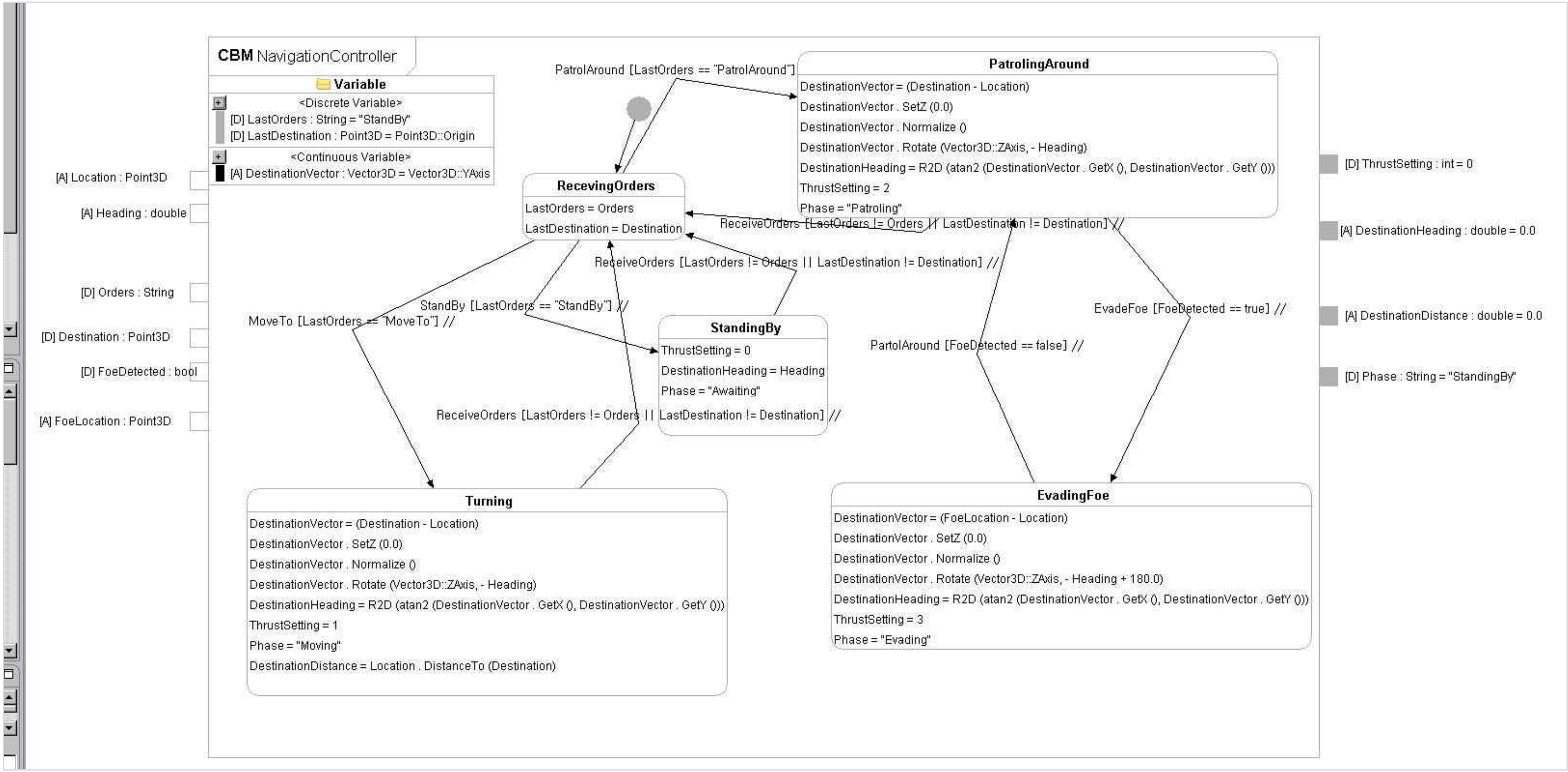
ECML

- ETRI CPS Modeling Language
  - Proposed by ETRI (Electronics and Telecommunication Research Institute) in Korea, 2011
  - Supporting ECML Modeling & Simulation
    - EcoPOD
    - EcoSIM
  - Refers to CHARON
  - Extends DEV&DESS formalism
  - Includes several syntactic sugar
    - 3 types of I/O : Discrete / Continuous / Event
    - Easy to model discrete systems as well as continuous systems
      - Allows to use 'phases' in addition to states $S = S^C \times S^D$
      - Allows to produce outputs by discrete transitions in addition to continuous/internal transitions
    - Not allow hierarchical state modeling as CHARON and Statecharts

EcoPOD


CPS - AAAA/Domain_Object_Model/ttt/Diagram/GroundVehicle.domain - CPS
File Edit Navigate Search Project Run Window Help
100%
CPS
AAAA.cps
ActuatingPart
GroundVehicle
Computation
ThrustController
SteeringController
Dynamics
Engine
Ground
BMP2
K-2
Structu
CSM GroundVehicle
[A] Location : Point3D
[A] EulerAngles : Angle3D
[A] Velocity : double
[A] NearestFoeDistance : double
[A] NearestFoeLocation : Point3D
[A] NearestFriendDistance : double
[A] NearestFreindLocation : Point3D
[D] IncomingOrders : String
[D] IncomingCoordinates : Point3D
CSM Sensing
[A] Location : Point3D
[A] EulerAnglers : Angle3D
[A] Velocity : double
[A] NearestFoeDistance : double
[A] NearestFoeLocation : Point3D
[A] MeasuredLocation : Point3D
[A] MeasuredHeading : double
[A] MeasuredVelocity : double
[D] FoeDetected : bool
[A] FoeLocation : Point3D
CSM Computation
[A] Location : Point3D
[A] Heading : double
[A] Velocity : double
[D] FoeDetected : bool
[A] FoeLocation : Point3D
[D] Orders : String
[D] Orders : String
[D] Destination : Point3D
[D] Destination : Point3D
[D] ThrustControl : double
[D] SteeringControl : int
[D] OutgoingOrders : String
CSM Communication
D] IncomingOrders : String
D] IncomingCoordinates : Point3D
CSM ActuatingPart
[D] ThrustControl : double
[D] SteeringControl : int
[A] Thrust : double
[A] AngularVelocities : Angle3D
CSM Dynamics
[A] Mass : double
[A] CdA : double
CBM Group
[C] Side: String
[C] Team : String
[A] Force : double
[A] AngularVelocities : Angle3D
[A] Mass : double
[A] CdA : double
[D] OutgoingDestinationTeam : String
[D] OutgoingOrders : String
[D] OutgoingCoordinates : Point3D
[C] Side : String
[C] Team : String
[D] Energy: int

ECML CMD (Coupled Model Diagram)



CBM NavigationController

Variable

<Discrete Variable>

[D] LastOrders : String = "StandBy"

[D] LastDestination : Point3D = Point3D::Origin

<Continuous Variable>

[A] DestinationVector : Vector3D = Vector3D::YAxis

[A] Location : Point3D

[A] Heading : double

[D] Orders : String

[D] Destination : Point3D

[D] FoeDetected : bool

[A] FoeLocation : Point3D

[D] ThrustSetting : int = 0

[A] DestinationHeading : double = 0.0

[A] DestinationDistance : double = 0.0

[D] Phase : String = "StandingBy"

RecevingOrders

LastOrders = Orders

LastDestination = Destination

PatrolingAround

DestinationVector = (Destination - Location)

DestinationVector . SetZ (0.0)

DestinationVector . Normalize ()

DestinationVector . Rotate (Vector3D::ZAxis, - Heading)

DestinationHeading = R2D (atan2 (DestinationVector . GetX (), DestinationVector . GetY ()))

ThrustSetting = 2

Phase = "Patroling"

StandingBy

ThrustSetting = 0

DestinationHeading = Heading

Phase = "Awaiting"

Turning

DestinationVector = (Destination - Location)

DestinationVector . SetZ (0.0)

DestinationVector . Normalize ()

DestinationVector . Rotate (Vector3D::ZAxis, - Heading)

DestinationHeading = R2D (atan2 (DestinationVector . GetX (), DestinationVector . GetY ()))

ThrustSetting = 1

Phase = "Moving"

DestinationDistance = Location . DistanceTo (Destination)

EvadingFoe

DestinationVector = (FoeLocation - Location)

DestinationVector . SetZ (0.0)

DestinationVector . Normalize ()

DestinationVector . Rotate (Vector3D::ZAxis, - Heading + 180.0)

DestinationHeading = R2D (atan2 (DestinationVector . GetX (), DestinationVector . GetY ()))

ThrustSetting = 3

Phase = "Evading"

PatrolAround [LastOrders == "PatrolAround"]

ReceiveOrders [LastOrders != Orders || LastDestination != Destination] //

ReceiveOrders [LastOrders != Orders || LastDestination != Destination] //

StandBy [LastOrders == "StandBy"] //

MoveTo [LastOrders == "MoveTo"] //

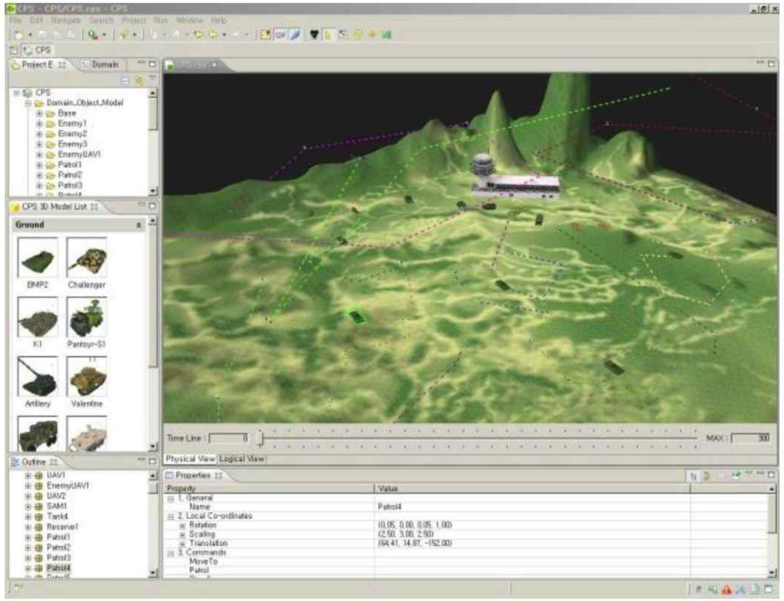
ReceiveOrders [LastOrders != Orders || LastDestination != Destination] //

EvadeFoe [FoeDetected == true] //

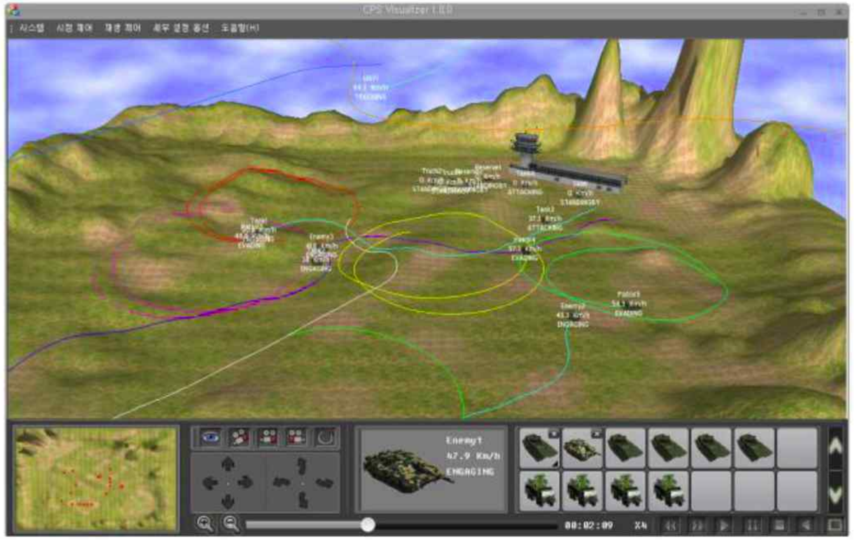
PartolAround [FoeDetected == false] //

ECML BMD (Basic Model Diagram)

EcoPOD



Property	Value
1. General	
Name	Patrol4
2. Local Co-ordinates	
Rotation	(0.05, 0.00, 0.05, 1.00)
Scaling	(2.50, 3.00, 2.50)
Translation	(64.41, 14.87, -152.00)
3. Commands	
MoveTo	
Patrol	



EcoSIM

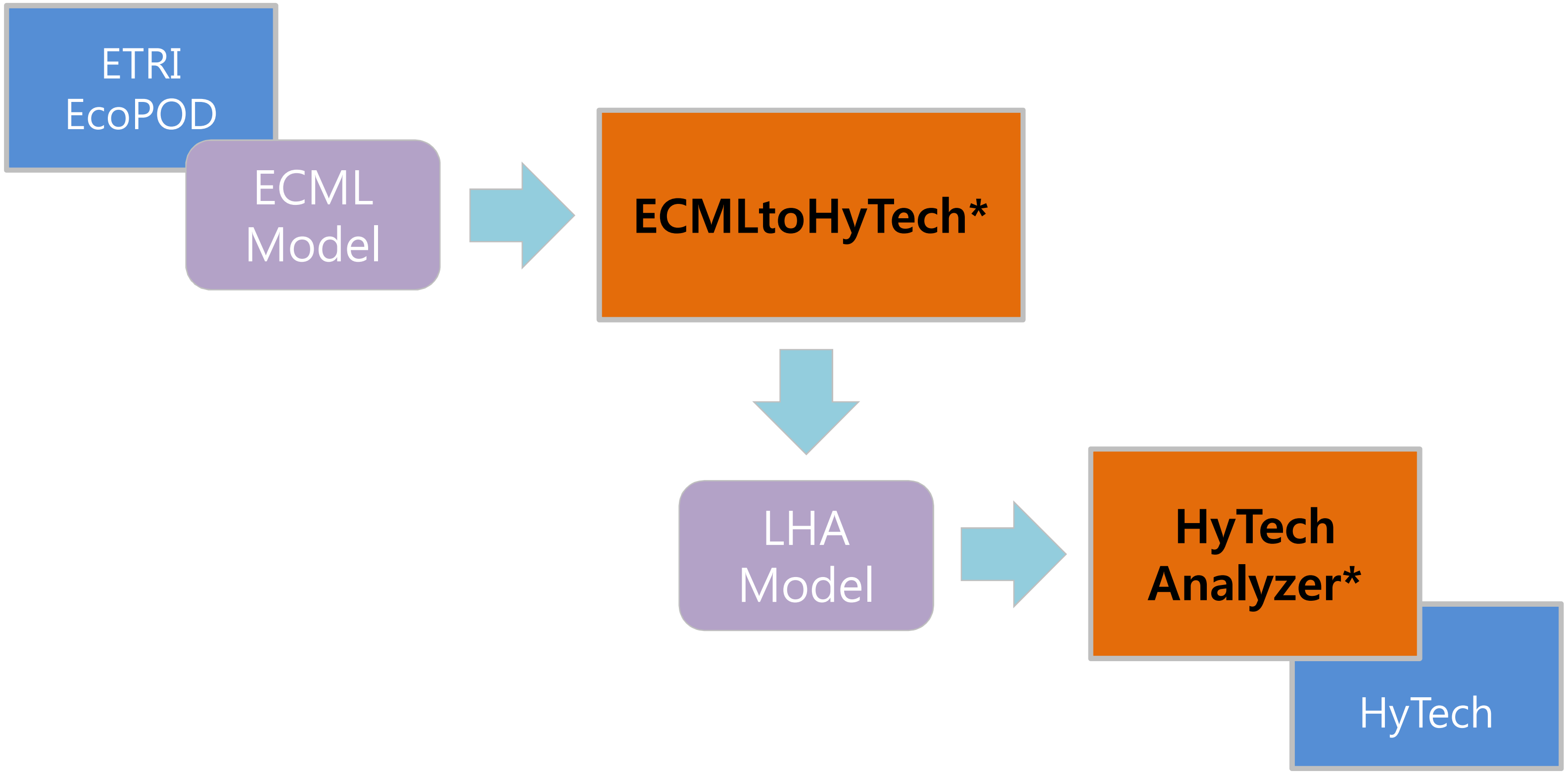
HyTech

- A basic verification tool for hybrid systems
  - Model checker
    - Safety verification , Parametric analysis
    - Simulation
  - Input-front-end: linear hybrid automata

  - No the concept of I/O variables
  - No GUI
  - No graphical editor for input programs

- We chose HyTech since it is the most fundamental model checker for hybrid automata.
  - Planning to use PHAVer and SpaceEX as well as HyTech

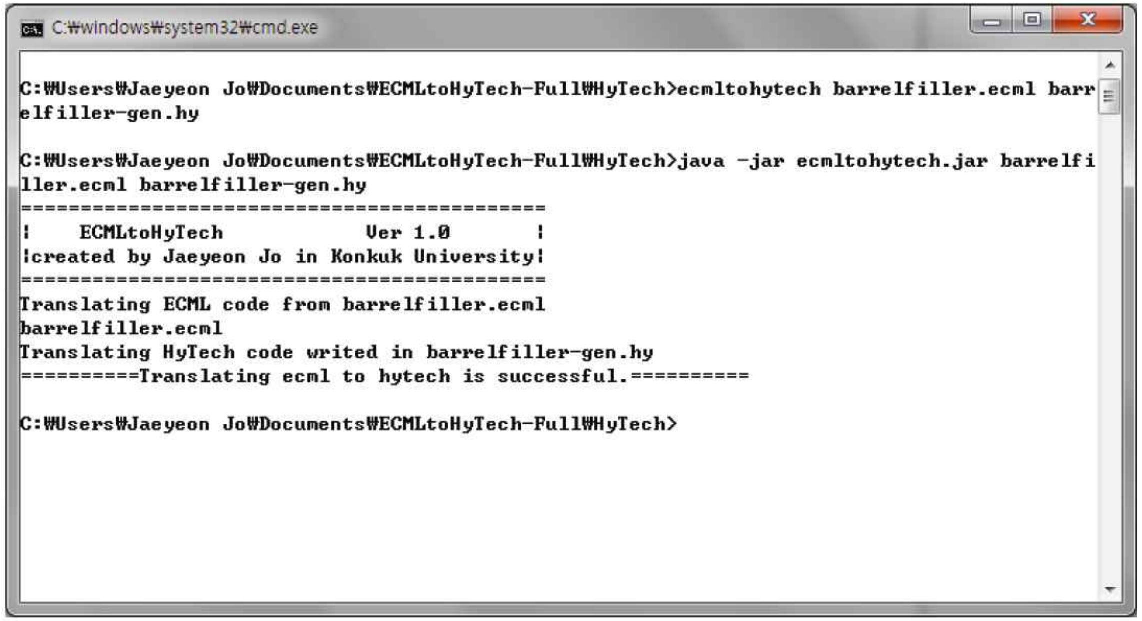
Formal Verification of ECML using HyTech



LHA: Linear Hybrid Automata

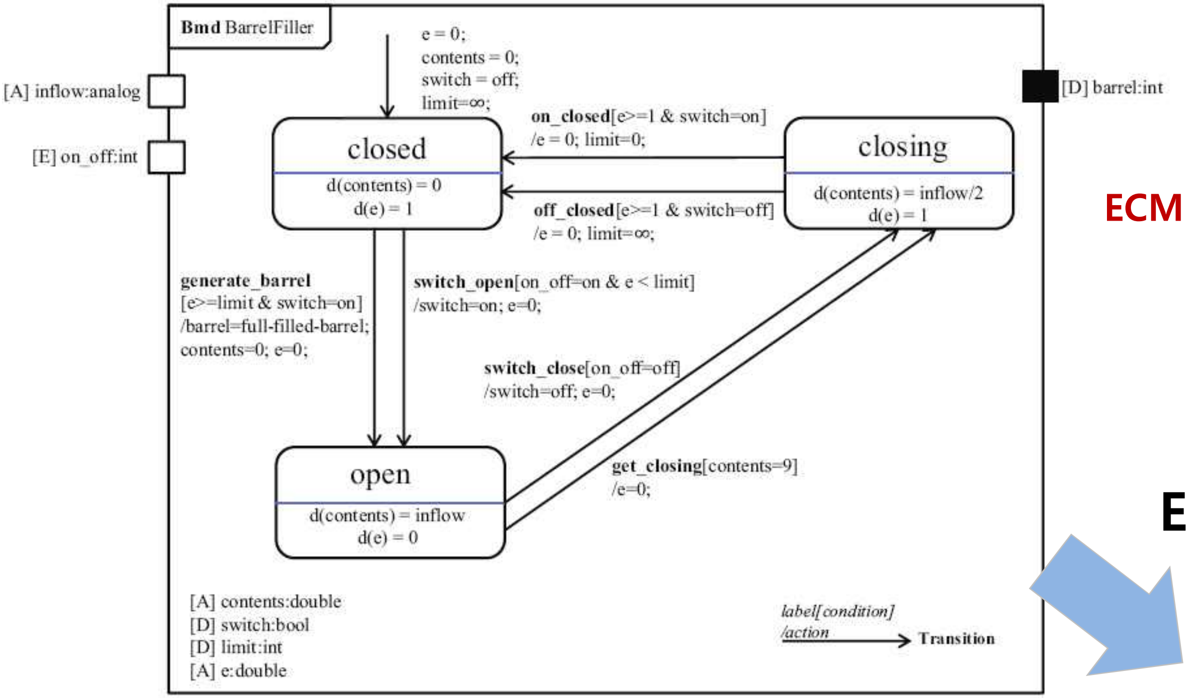
ECMLtoHyTech

- A mechanical translator from ECML to LHA
  - Defined translation rules semi-formally
  - Resolved semantic gap between ECML and LHA of HyTech
    - Uses I/O automaton additionally
    - Uses invariant conditions of LHA to enforce state transition



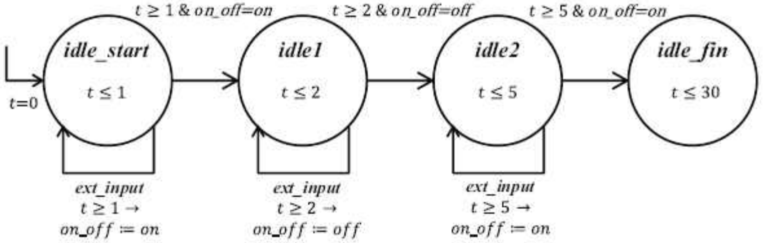
```
C:₩Users₩Jaeyeon Jo₩Documents₩ECMLtoHyTech-Full₩HyTech>ecmltohytech barrelfiller.ecml barr
elfiller-gen.hy

C:₩Users₩Jaeyeon Jo₩Documents₩ECMLtoHyTech-Full₩HyTech>java -jar ecmltohytech.jar barrelfi
ller.ecml barrelfiller-gen.hy
============================================
|    ECMLtoHyTech            Ver 1.0        |
|created by Jaeyeon Jo in Konkuk University|
============================================
Translating ECML code from barrelfiller.ecml
barrelfiller.ecml
Translating HyTech code writed in barrelfiller-gen.hy
==========Translating ecml to hytech is successful.==========

C:₩Users₩Jaeyeon Jo₩Documents₩ECMLtoHyTech-Full₩HyTech>
```

ECMLtoHyTech



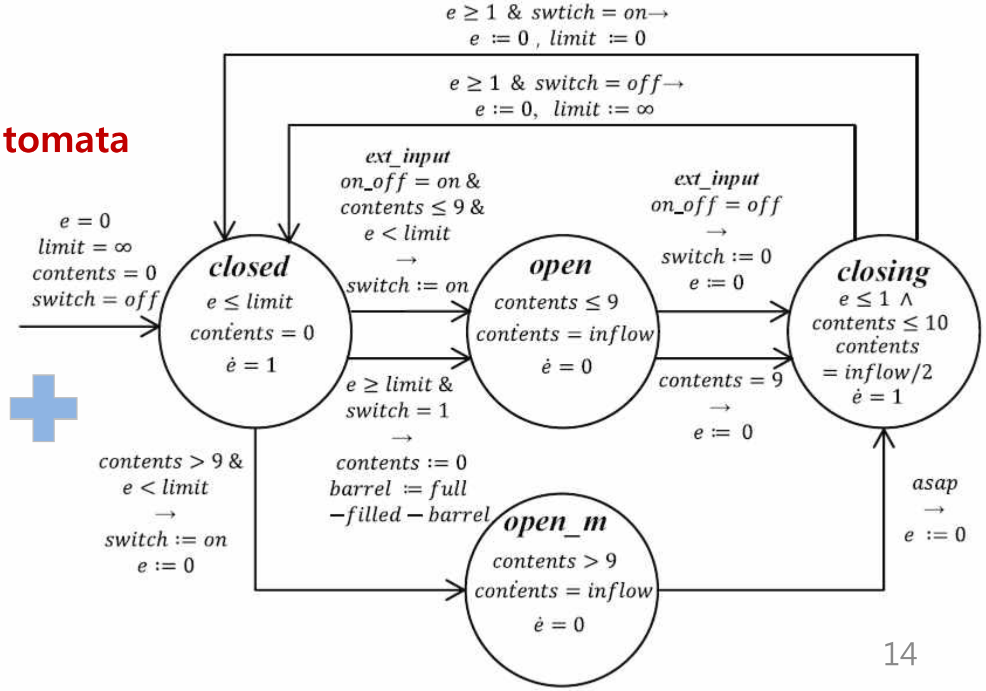
ECML BMD

ECMLtoHyTech*

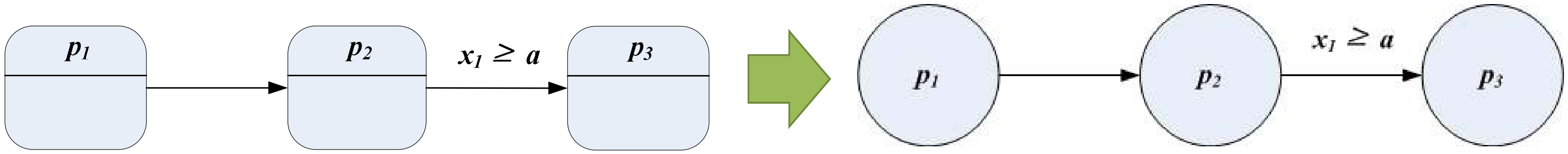


(I/O Automaton)

Linear Hybrid Automata

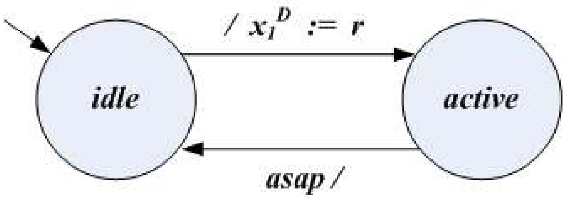


Translation Rules



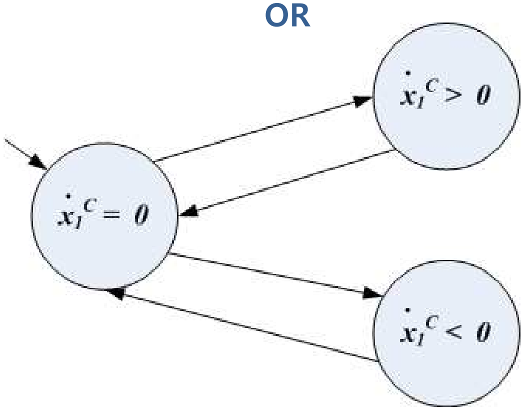
Translation of Transitions

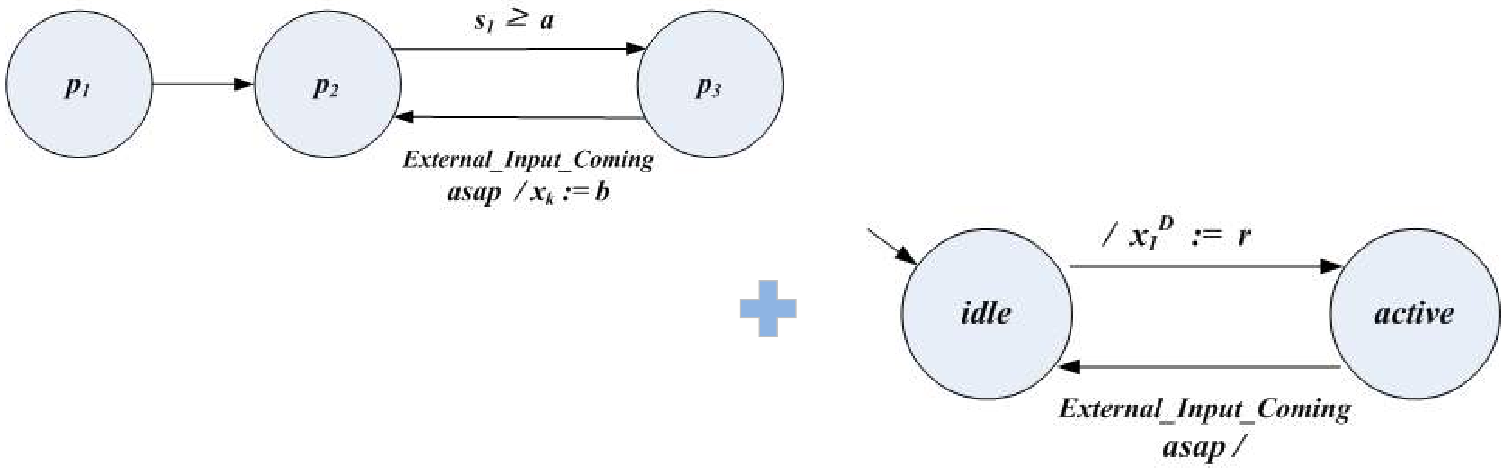
Discrete input automaton



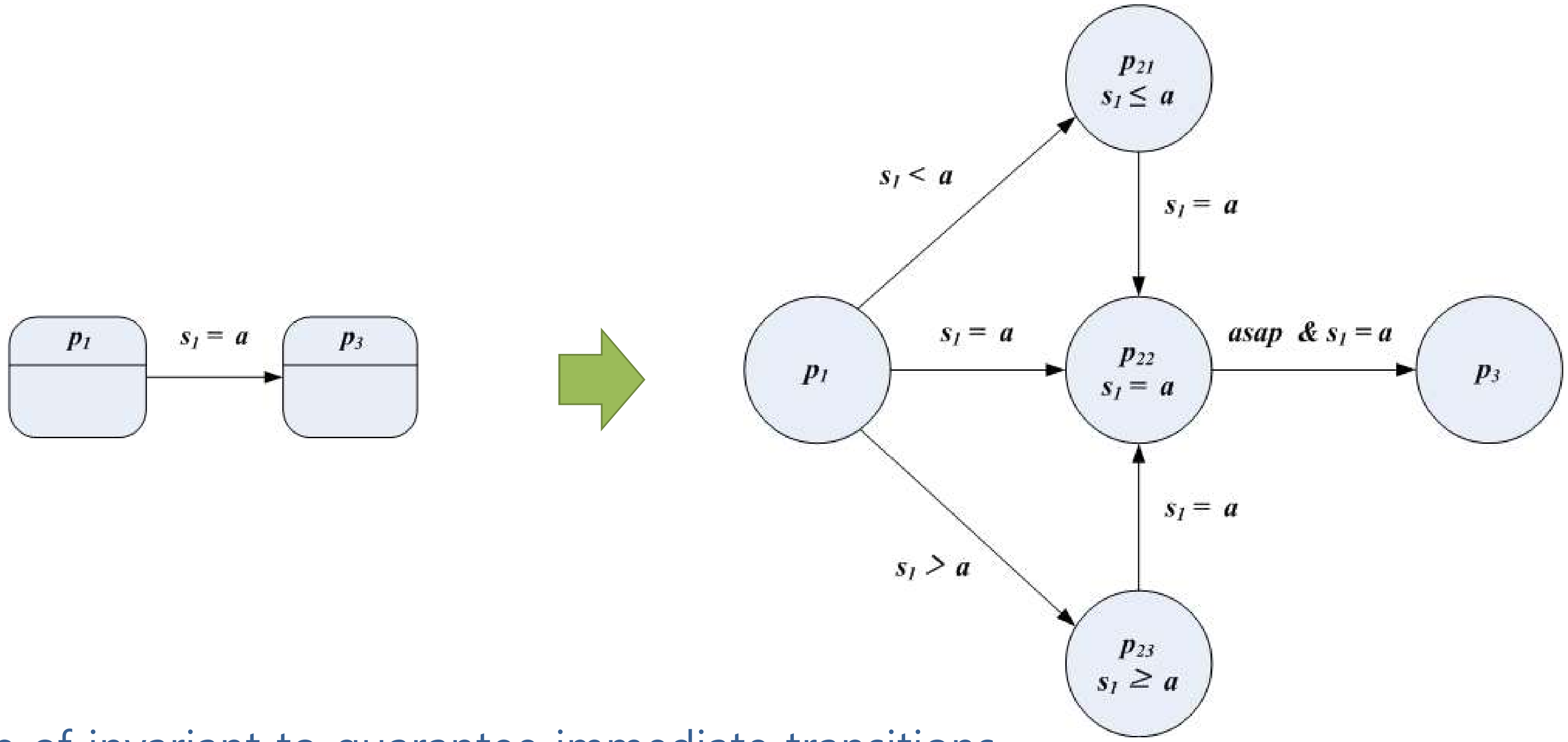
OR

Continuous input automaton





Use synchronized labels to model immediate coming of discrete inputs



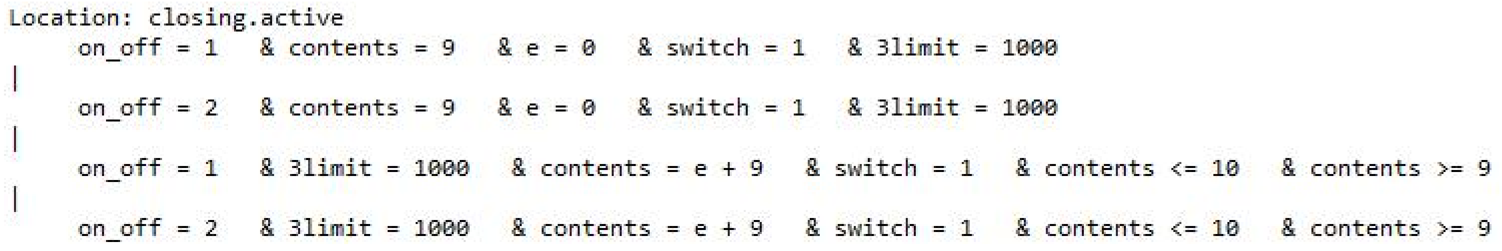
Use of invariant to guarantee immediate transitions

HyTech Analyzer

- A visual assistant of HyTech
  - Eclipse plug-in
  - Read LHA, execute HyTech, and visualize verification results
  - Supporting
    - RegionTableViewer
    - RegionAnalyzer
    - TraceTableViewer
    - TraceChart

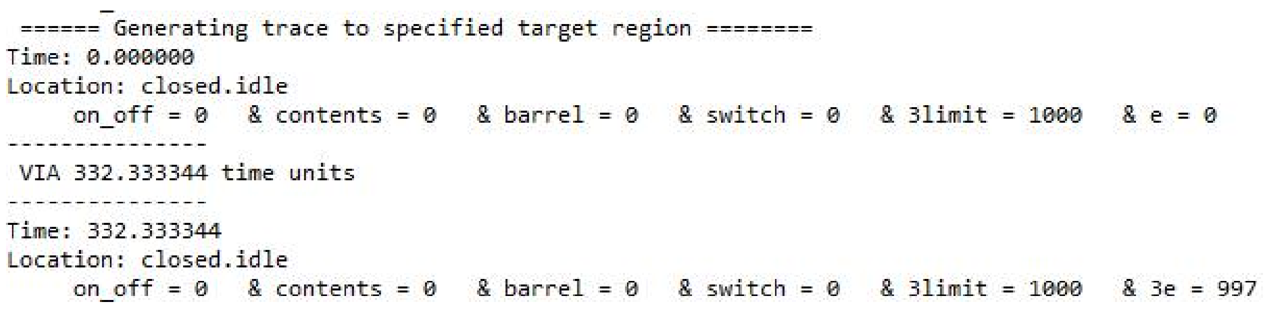
HyTech Outputs

Region

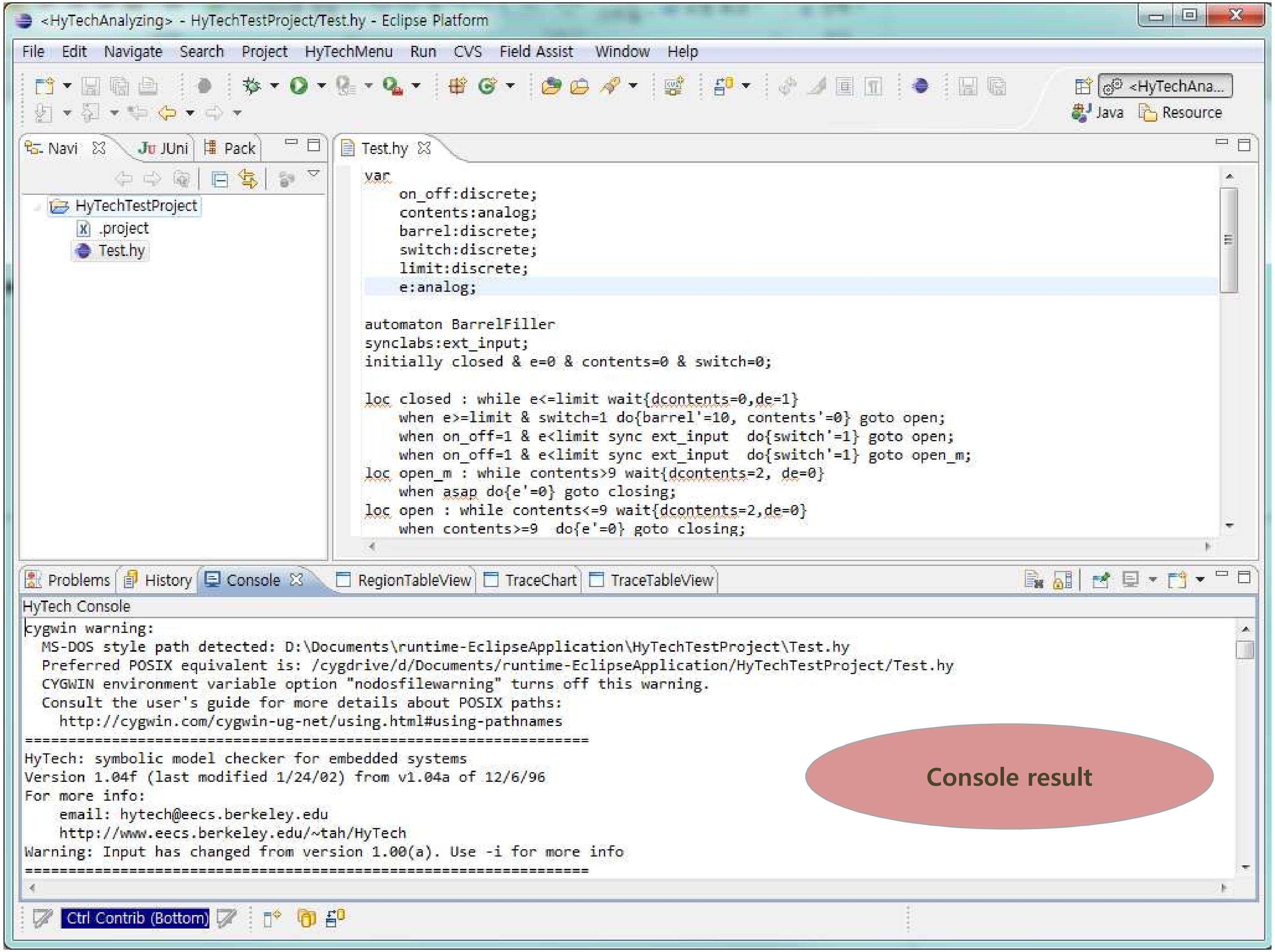


```
Location: closing.active
     on_off = 1   & contents = 9   & e = 0   & switch = 1   & 3limit = 1000
|
     on_off = 2   & contents = 9   & e = 0   & switch = 1   & 3limit = 1000
|
     on_off = 1   & 3limit = 1000   & contents = e + 9   & switch = 1   & contents <= 10   & contents >= 9
|
     on_off = 2   & 3limit = 1000   & contents = e + 9   & switch = 1   & contents <= 10   & contents >= 9
```

Trace



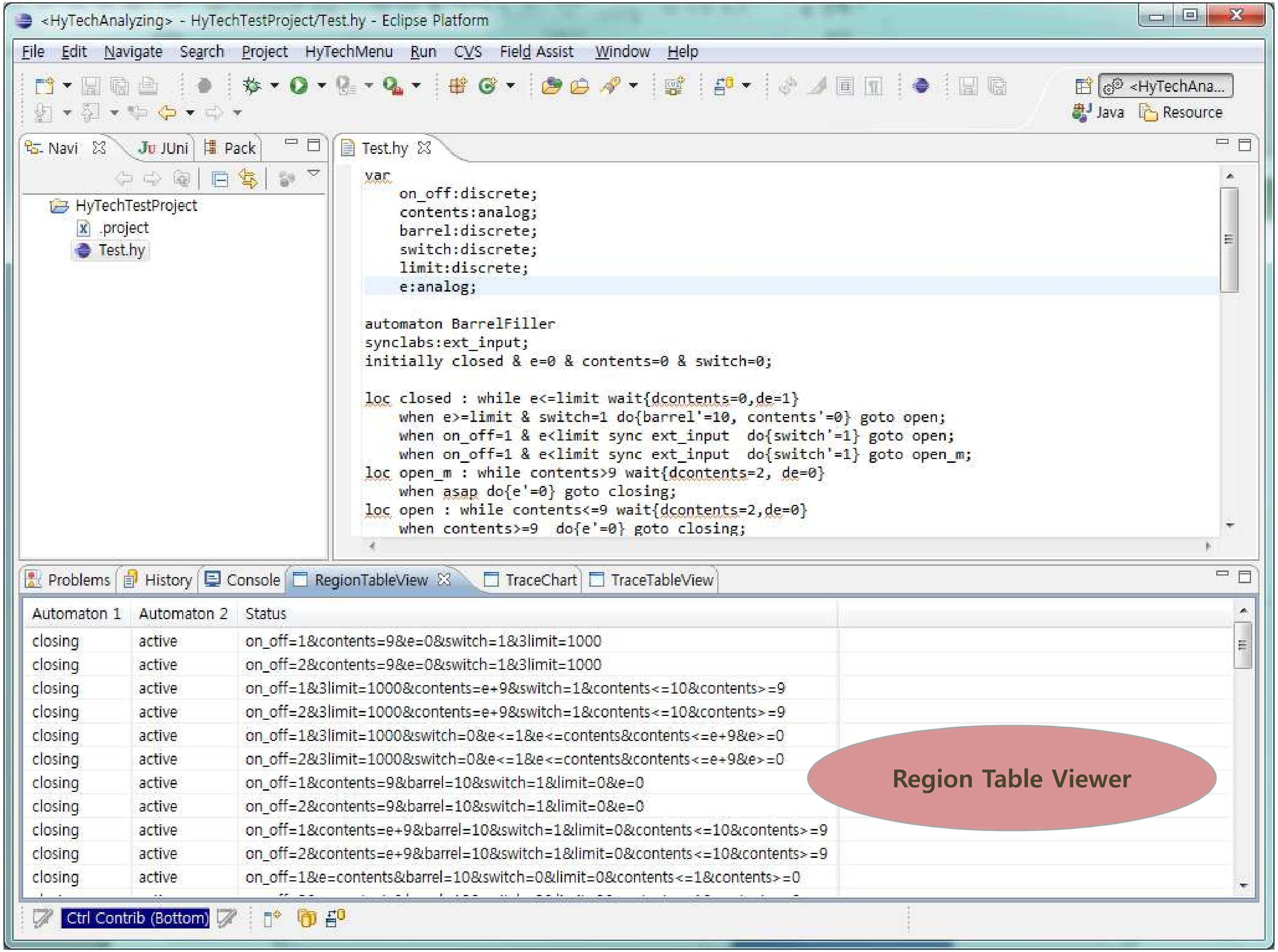
```
 ====== Generating trace to specified target region ========
Time: 0.000000
Location: closed.idle
     on_off = 0   & contents = 0   & barrel = 0   & switch = 0   & 3limit = 1000   & e = 0
---------------
 VIA 332.333344 time units
---------------
Time: 332.333344
Location: closed.idle
     on_off = 0   & contents = 0   & barrel = 0   & switch = 0   & 3limit = 1000   & 3e = 997
```



```
var
    on_off:discrete;
    contents:analog;
    barrel:discrete;
    switch:discrete;
    limit:discrete;
    e:analog;

automaton BarrelFiller
synclabs:ext_input;
initially closed & e=0 & contents=0 & switch=0;

loc closed : while e<=limit wait{dcontents=0,de=1}
    when e>=limit & switch=1 do{barrel'=10, contents'=0} goto open;
    when on_off=1 & e<limit sync ext_input  do{switch'=1} goto open;
    when on_off=1 & e<limit sync ext_input  do{switch'=1} goto open_m;
loc open_m : while contents>9 wait{dcontents=2, de=0}
    when asap do{e'=0} goto closing;
loc open : while contents<=9 wait{dcontents=2,de=0}
    when contents>=9  do{e'=0} goto closing;
```

```
cygwin warning:
  MS-DOS style path detected: D:\Documents\runtime-EclipseApplication\HyTechTestProject\Test.hy
  Preferred POSIX equivalent is: /cygdrive/d/Documents/runtime-EclipseApplication/HyTechTestProject/Test.hy
  CYGWIN environment variable option "nodosfilewarning" turns off this warning.
  Consult the user's guide for more details about POSIX paths:
    http://cygwin.com/cygwin-ug-net/using.html#using-pathnames
=================================================================
HyTech: symbolic model checker for embedded systems
Version 1.04f (last modified 1/24/02) from v1.04a of 12/6/96
For more info:
    email: hytech@eecs.berkeley.edu
    http://www.eecs.berkeley.edu/~tah/HyTech
Warning: Input has changed from version 1.00(a). Use -i for more info
=================================================================
```

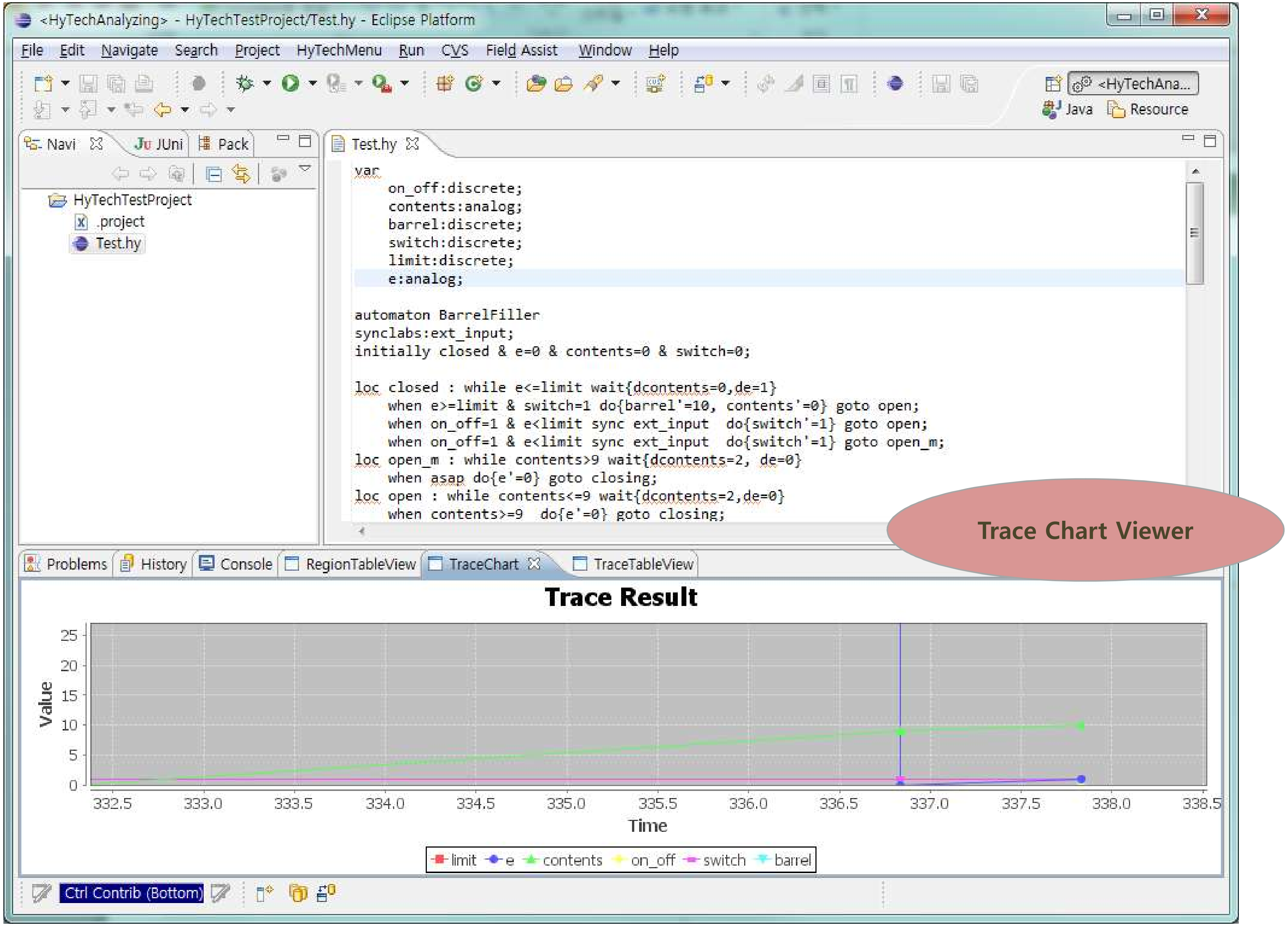
Console result



```
var
    on_off:discrete;
    contents:analog;
    barrel:discrete;
    switch:discrete;
    limit:discrete;
    e:analog;

automaton BarrelFiller
synclabs:ext_input;
initially closed & e=0 & contents=0 & switch=0;

loc closed : while e<=limit wait{dcontents=0,de=1}
    when e>=limit & switch=1 do{barrel'=10, contents'=0} goto open;
    when on_off=1 & e<limit sync ext_input  do{switch'=1} goto open;
    when on_off=1 & e<limit sync ext_input  do{switch'=1} goto open_m;
loc open_m : while contents>9 wait{dcontents=2, de=0}
    when asap do{e'=0} goto closing;
loc open : while contents<=9 wait{dcontents=2,de=0}
    when contents>=9  do{e'=0} goto closing;
```

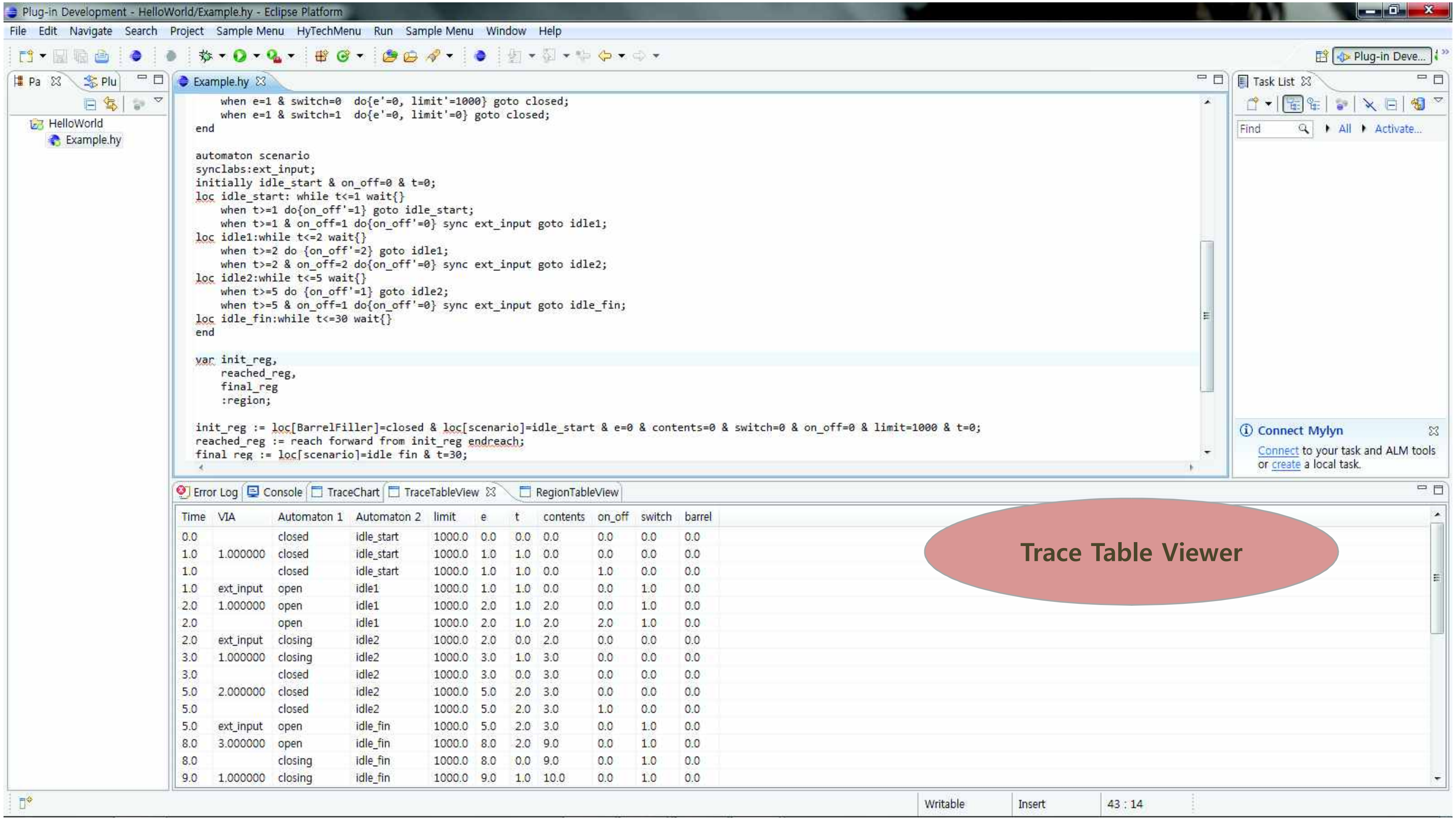
Automaton 1	Automaton 2	Status
closing	active	on_off=1&contents=9&e=0&switch=1&3limit=1000
closing	active	on_off=2&contents=9&e=0&switch=1&3limit=1000
closing	active	on_off=1&3limit=1000&contents=e+9&switch=1&contents<=10&contents>=9
closing	active	on_off=2&3limit=1000&contents=e+9&switch=1&contents<=10&contents>=9
closing	active	on_off=1&3limit=1000&switch=0&e<=1&e<=contents&contents<=e+9&e>=0
closing	active	on_off=2&3limit=1000&switch=0&e<=1&e<=contents&contents<=e+9&e>=0
closing	active	on_off=1&contents=9&barrel=10&switch=1&limit=0&e=0
closing	active	on_off=2&contents=9&barrel=10&switch=1&limit=0&e=0
closing	active	on_off=1&contents=e+9&barrel=10&switch=1&limit=0&contents<=10&contents>=9
closing	active	on_off=2&contents=e+9&barrel=10&switch=1&limit=0&contents<=10&contents>=9
closing	active	on_off=1&e=contents&barrel=10&switch=0&limit=0&contents<=1&contents>=0

Region Table Viewer



```
var
    on_off:discrete;
    contents:analog;
    barrel:discrete;
    switch:discrete;
    limit:discrete;
    e:analog;

automaton BarrelFiller
synclabs:ext_input;
initially closed & e=0 & contents=0 & switch=0;

loc closed : while e<=limit wait{dcontents=0,de=1}
    when e>=limit & switch=1 do{barrel'=10, contents'=0} goto open;
    when on_off=1 & e<limit sync ext_input  do{switch'=1} goto open;
    when on_off=1 & e<limit sync ext_input  do{switch'=1} goto open_m;
loc open_m : while contents>9 wait{dcontents=2, de=0}
    when asap do{e'=0} goto closing;
loc open : while contents<=9 wait{dcontents=2,de=0}
    when contents>=9  do{e'=0} goto closing;
```

Trace Chart Viewer


Plug-in Development - HelloWorld/Example.hy - Eclipse Platform
File Edit Navigate Search Project Sample Menu HyTechMenu Run Sample Menu Window Help
Plug-in Deve...
Pa
Plu
HelloWorld
Example.hy
Task List
Find
All
Activate...
Connect Mylyn
Connect to your task and ALM tools or create a local task.
Error Log
Console
TraceChart
TraceTableView
RegionTableView
Writable
Insert
43 : 14

```
    when e=1 & switch=0  do{e'=0, limit'=1000} goto closed;
    when e=1 & switch=1  do{e'=0, limit'=0} goto closed;
end

automaton scenario
synclabs:ext_input;
initially idle_start & on_off=0 & t=0;
loc idle_start: while t<=1 wait{}
    when t>=1 do{on_off'=1} goto idle_start;
    when t>=1 & on_off=1 do{on_off'=0} sync ext_input goto idle1;
loc idle1:while t<=2 wait{}
    when t>=2 do {on_off'=2} goto idle1;
    when t>=2 & on_off=2 do{on_off'=0} sync ext_input goto idle2;
loc idle2:while t<=5 wait{}
    when t>=5 do {on_off'=1} goto idle2;
    when t>=5 & on_off=1 do{on_off'=0} sync ext_input goto idle_fin;
loc idle_fin:while t<=30 wait{}
end

var init_reg,
    reached_reg,
    final_reg
    :region;

init_reg := loc[BarrelFiller]=closed & loc[scenario]=idle_start & e=0 & contents=0 & switch=0 & on_off=0 & limit=1000 & t=0;
reached_reg := reach forward from init_reg endreach;
final reg := loc[scenario]=idle fin & t=30;
```

Time	VIA	Automaton 1	Automaton 2	limit	e	t	contents	on_off	switch	barrel
0.0		closed	idle_start	1000.0	0.0	0.0	0.0	0.0	0.0	0.0
1.0	1.000000	closed	idle_start	1000.0	1.0	1.0	0.0	0.0	0.0	0.0
1.0		closed	idle_start	1000.0	1.0	1.0	0.0	1.0	0.0	0.0
1.0	ext_input	open	idle1	1000.0	1.0	1.0	0.0	0.0	1.0	0.0
2.0	1.000000	open	idle1	1000.0	2.0	1.0	2.0	0.0	1.0	0.0
2.0		open	idle1	1000.0	2.0	1.0	2.0	2.0	1.0	0.0
2.0	ext_input	closing	idle2	1000.0	2.0	0.0	2.0	0.0	0.0	0.0
3.0	1.000000	closing	idle2	1000.0	3.0	1.0	3.0	0.0	0.0	0.0
3.0		closed	idle2	1000.0	3.0	0.0	3.0	0.0	0.0	0.0
5.0	2.000000	closed	idle2	1000.0	5.0	2.0	3.0	0.0	0.0	0.0
5.0		closed	idle2	1000.0	5.0	2.0	3.0	1.0	0.0	0.0
5.0	ext_input	open	idle_fin	1000.0	5.0	2.0	3.0	0.0	1.0	0.0
8.0	3.000000	open	idle_fin	1000.0	8.0	2.0	9.0	0.0	1.0	0.0
8.0		closing	idle_fin	1000.0	8.0	0.0	9.0	0.0	1.0	0.0
9.0	1.000000	closing	idle_fin	1000.0	9.0	1.0	10.0	0.0	1.0	0.0

Trace Table Viewer

Conclusion and Future Work

- We have been trying to verify ECML models using HyTech
  - Have developed
    - Translation rules from ECML to linear hybrid automata
    - A mechanical translator – ECMLtoHyTech
    - A visual assistant of HyTech – HyTech Analyzer
  - Also found problems
    - Semantic gap between ECML and LHA
    - Limitation of the HyTech verification
    - Restriction on modeling by linear hybrid automata
- We are now trying SpaceEx.