TOWARDS
Cluster in the Loop Simulation Framework based on Formal Model-based Testing for Embedded Network Systems

Eun-Hye Choi, Yoshiki Kinoshita, Hiroyuki Ozaki, Hayao Nakahara, AIST/CVS

Masahiro Aoki, Keiichi Yoshisaka, Hiroshi Mine, Daikin Industries, Ltd.

Toru Shimizu, Renesas Technology Corp.

Outline

- Backgrounds
  - Embedded network systems
  - Difficulties in V&V of embedded network systems
- Our testing framework
- Design of the proposed testing system

Embedded Network Systems

- Embedded systems all around us:
  - In-vehicle systems
  - Consumer electronics
  - Building and energy management systems.

  Nowaday, they even form *networks*, which of course increase the scale and complexity a lot.

Verification and validation of embedded network systems wanted!

Outline

- Backgrounds
- Our testing framework
  - Our project
  - Target system
  - Current status of testing the target system
  - Key concept of our testing framework
- Design of the proposed testing system

Our Project

To develop a system to validate network system constituting air conditioners, especially for a large building.

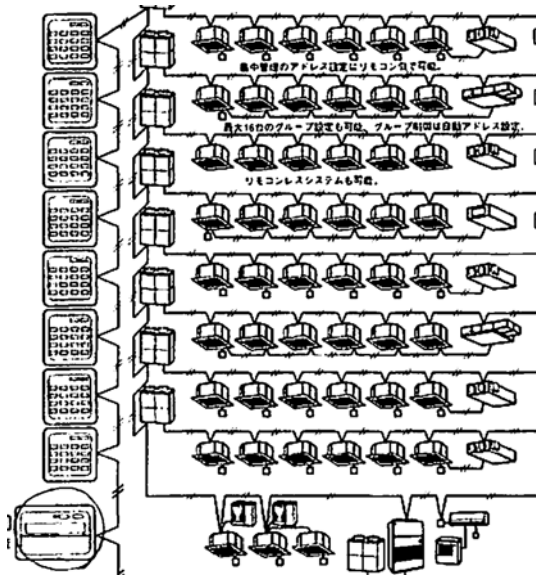
Currently: feasibility study being done in collaboration with

- Daikin industries, ltd. (D): 
- developing air-conditioning systems and building energy management systems.
- Renesas Technology Corp. (R): RENESAS Everywhere you imagine.
- developing embedded micro processing units and related development tools.

Target System

Air-conditioner network for buildings developed by D.

- The target system consists of
  - controllers,
  - indoor/outdoor equipments including actuators
  - sensors, etc.
- Take the system construction where all MPUs are of R Ltd.
- Hundreds of nodes may be connected in the target system.



Air-conditioning Network System

Current status of testing

Current status of testing

- Specifications written in natural language, tables, and state transition diagrams.
- Manual generation of test cases.
- Manual testing using real equipments.

→ More testing is seeked for in validation of huge complex systems.

→ New testing method wanted.

New method for more test cases!

Current status of testing:

- Specifications written in natural language, tables, and state transition diagrams.
- Manual generation of test cases.
- Manual testing using real equipments.

- Testing for many more test cases preferrable
- New testing method wanted.

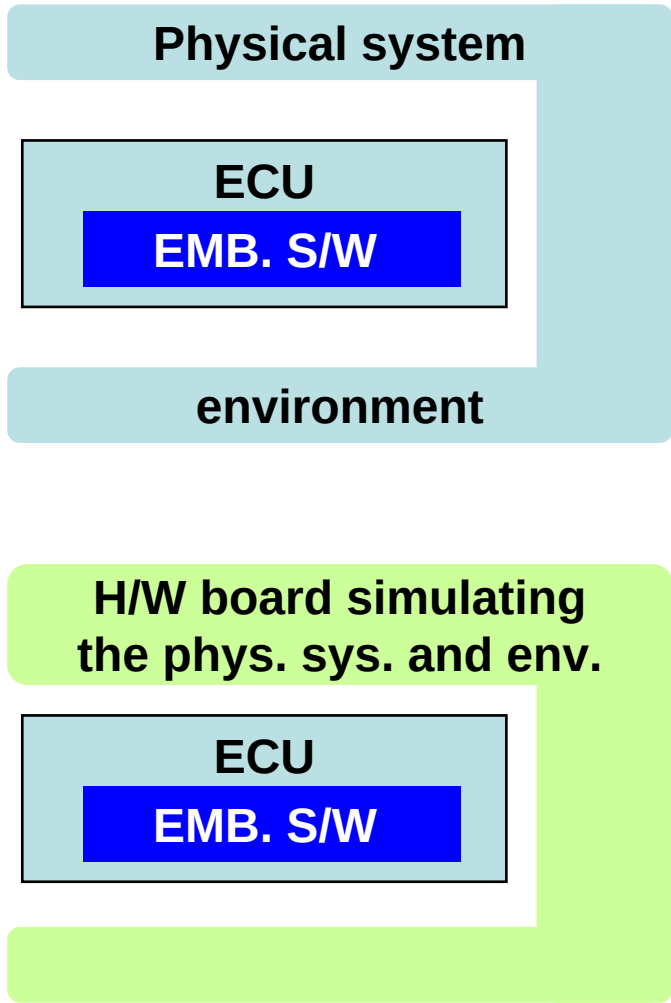
Two keys in our proposal

- Two Keys
  - Cluster in the Loop Simulation (CILS)
  - Formal Model-based Testing

Hardware in the Loop Simulation (HILS)

- HILS simulates the behavior of
  - a physical system and
  - an environment

  with which electronic control units of the embedded system interacts
- In HILS, both simulators and real equipments are used for testing.
  - helpful for testing complex real-time embedded sys.
  - Not suitable for testing large scale embedded network systems.

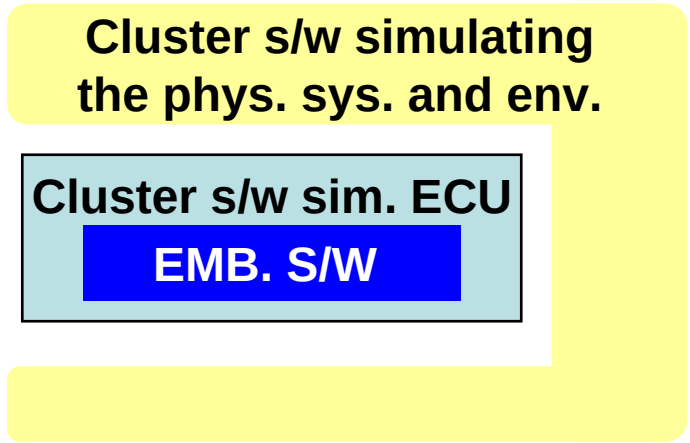


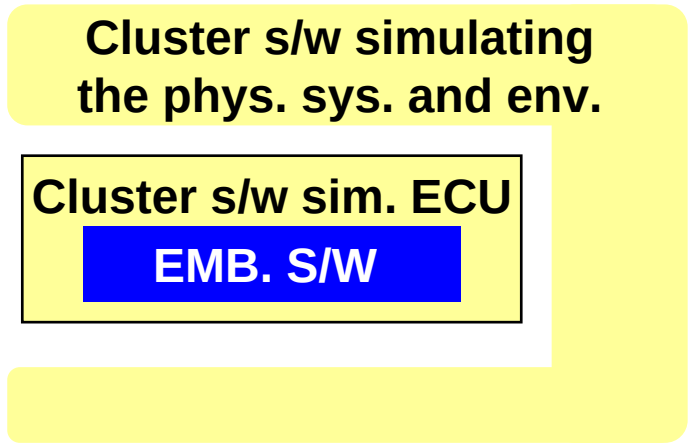
Cluster in the Loop Simulation (CILS)

- HILS uses hardware board; *CILS* uses a cluster instead.
- that simulates not only the physical system and the environment but also electronic control units to be connected them in a cluster computing system.

→ The whole embedded network system is simulated as software in a cluster system.

→ Real equipments are not necessary for testing.





CILS vs. HILS

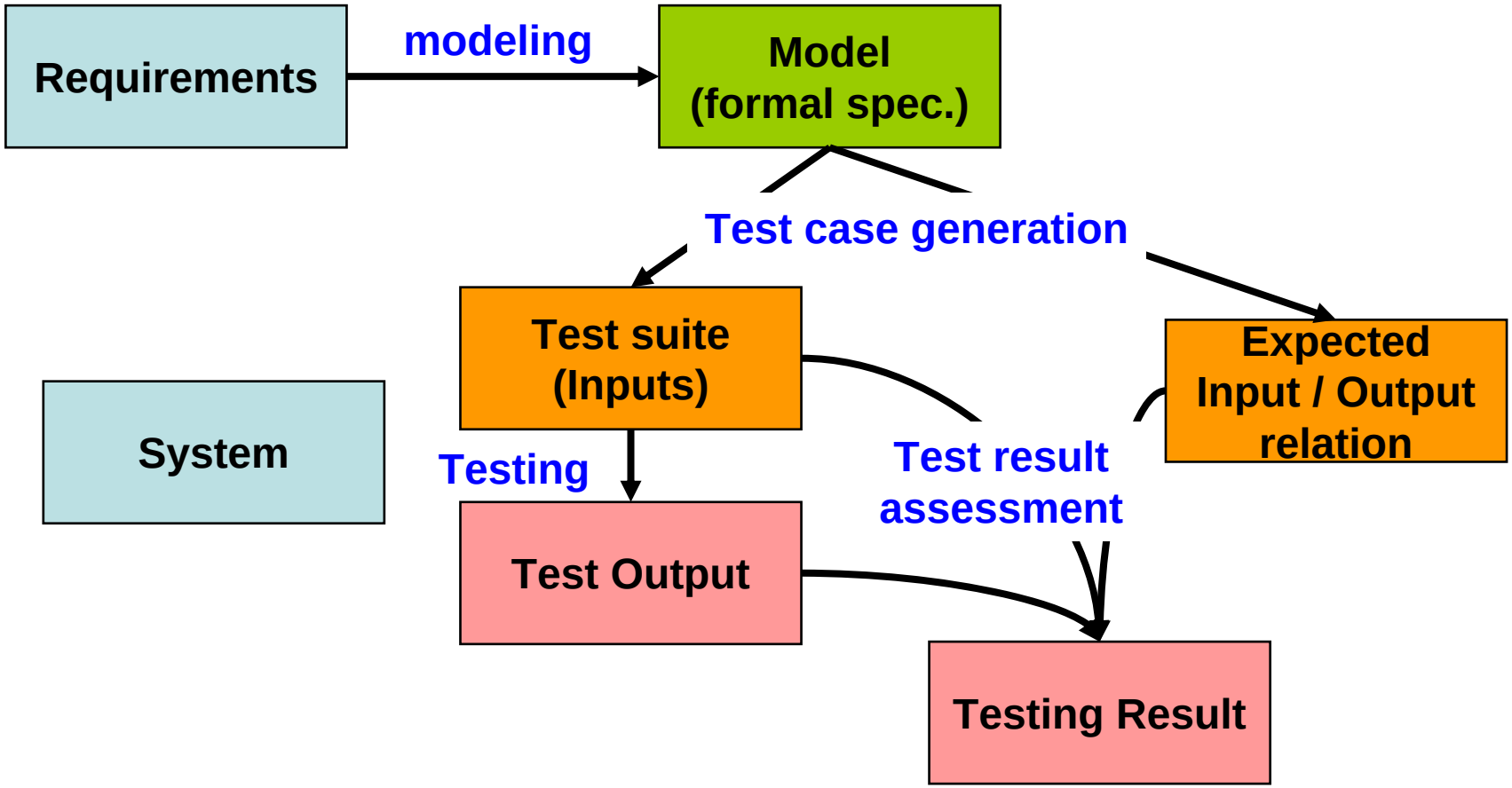
	CILS	HILS
Application in design phase	☺	☹
Data processing	☺	☹
Real-time testing	☹	☺
Cost for testing	☺	☹
Time for testing	☺	☹
Scalability	☺	☹
Modification of network topology	☺	☹
Automatic confirmation of network configuration	☺	☹

Outline

- Backgrounds
- Our testing framework
  - Target system
  - Previous testing of the target system
  - Key concept of our testing framework
    - CILS
    - Formal Model-based Testing
- Design of the proposed testing system

Model-based Testing

Generating a test suite from a formal specification (model).



Model-based Testing

- There are many different ways
  - What kind of model description is supposed?
    - Finite State Machines: Decision tables, State charts..
  - How a test suite is derived?
    - Searching for execution traces in an abstract model,
    - Finding test cases by constraint programming,
    - Using counterexample paths by model checking,
    - Selecting test cases by partitioning a model to classes using theorem proving…

Our Approach

- Given informal specification written in Japanese etc.,
- Derive a formal specification (Model) written in Agda.
- Build a tool which automatically generates test suites for the specificaiton.
  - The specification must be in a form such that such automatic generation is possible.

A case study

Informal specification I

単位外部仕様書

単位外部 仕様名称 （機能分類）	凍結防止制御 （ 冷媒付加制御 ）
代表機種名 （PGG番号）	[redacted]
変更指示書番号 （機能仕様書番号）	
ベース単位外部仕様書 （累進含む）	PGG05B003
ベース仕様書からの 変更点概要	

2006年3月6日

[redacted]

1/2

1. 目的
冷房モード又は除湿モード時の、室内熱交氷結を回避するために、熱交温度が所定温度以下に下がりすぎないように、運転周波数を垂下させる。

2. 入出力情報

入力情報	出力情報
①圧縮機運転状態	①凍結防止ステータス
②室内熱交温度	②室内ファン目標回転数
③運転モード	
④室温	
⑤強制冷房指令	
⑥室外凍防状態	

3. 機能説明

(1)強制運転以外の制御内容
・要運転モードが冷房・ドライモード(*)時に凍結防止ステータスを決定する。
・圧縮機運転開始時よりタイマTTOを開は、プルダウンなどの過渡変化を考慮して、熱交温度下がっても凍結防止ステータスは復帰ゾーンとする。(TTO:凍防制御ガードタイマ)
・TTOは圧縮機運転停止でタイマクリアする。
・TTOオーバ後は、以下の熱交温度ゾーンによって凍結防止ステータスを決定する。

熱交温度ゾーン	凍結防止ステータス	
Aゾーン	復帰	
Bゾーン	ｱｸｾﾙ	
Cゾーン	無変化	
Dゾーン	垂下	
Eゾーン	txカウント中	垂下 ※1
	txカウントオーバー	停止 ※2

(*) ドライモードは除湿および
セレクトドライを指す

※1)
熱交温度ゾーンがEゾーン以外からEゾーンに変化しても、即氷結には至らないため、
熱交温度ゾーンD→Eの変化で一定時間(tx:運転モードによる変数)は垂下とする。
一定時間後、熱交温度ゾーンがEゾーンであれば停止とする。
・txは熱交温度ゾーンがD→Eに変化したとき、カウントを開始。
・txカウント中に熱交温度ゾーンがE→Dに変化したとき、カウント値をクリアする。
・txは運転モード毎に定数記号が異なる。

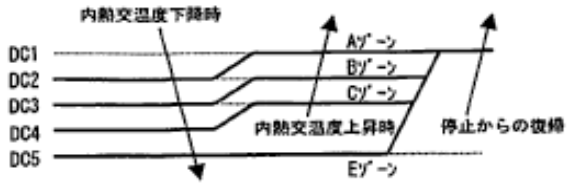
運転ﾓｰﾄﾞ	txの値
ﾄﾞﾗｲ1,2,3ﾓｰﾄﾞ中	TTOFGDRY
上記以外の冷房ﾓｰﾄﾞ中	TTOFG

※2)
一旦、凍結防止ステータスが停止になったら、熱交温度ゾーンがAゾーンになるまで停止とする。

Informal specification II

2/2

●熱交温度ゾーンについて



内熱交温度下降時	内熱交温度上昇時	熱交温度ゾーン
DC2≦熱交	DC1≦熱交	Aゾーン
DC3≦熱交<DC2	DC2≦熱交<DC1	Bゾーン
DC4≦熱交<DC3	DC3≦熱交<DC2	Cゾーン
DC5≦熱交<DC4	DC5≦熱交<DC3	Dゾーン
熱交<DC5	熱交<DC5	Eゾーン

運転モード	DC1	DC2	DC3	DC4	DC5
再熱ドライモード中	DTKTR4	DTKTR1 + DTKTR3	DTKTR1 + DTKTR2	DTKTR1	DTKTR0
上記以外の冷房モード中	DTO4	DTO1+DTO3	DTO1+DTO2	DTO1	DTO0

●凍結防止による圧縮機停止時の処理
凍防停止時、ファンを停止すると、ファンローター温度が上昇するスピードが遅くなる。一方、ローターの周囲温度は室温に近づくため、ローター結露が発生する可能性がある。また、室温が低いときにはファンOFFであると、放熱するのに時間がかかる。
以上より、凍結防止で停止(室外機より凍結停止を受けたとき)した場合、次回の圧縮機ON迄はﾌｧﾝ回転数の下限値をW2とする。

熱交温度ゾーン	凍結防止ステータス
Aゾーン	復帰
Bゾーン	ｱｯﾌﾟ
Cゾーン	無変化
Dゾーン	落下
Eゾーン	停止(一旦停止するかは室外からの指令による。)

(2)強制運転の制御内容
室内で設定する強制運転(本体運転SW5秒押し)と室外で設定する強制冷房運転の仕様を共通仕様にする。
・圧縮機運転開始時よりタイマTTOは、凍結防止ステータスは復帰ゾーンとする。
・TTOは圧縮機運転停止でタイマクリアする。
・TTOオーバ後は、以下の熱交温度ゾーンによって凍結防止ステータスを決定する。

4 定数

名称	説明	単位	刻み	変化範囲	狙い値
TTO	CompON時の凍防制御ﾏｽｸﾀｲﾏ	sec	10	10~630	240
TTOFG	凍防ステータス遷移ﾀｲﾏ(再熱ﾄﾞﾗｲ,けつろ防止以外のﾓｰﾄﾞ)	sec	10	10~630	180
TTOFGKTR	凍防ステータス遷移ﾀｲﾏ(けつろ防止用)	sec	10	10~630	60
TTOFGDRY	凍防ステータス遷移ﾀｲﾏ(再熱除湿用)	sec	10	10~630	600
DTO0	凍防制御 停止温度	deg	0.5	-10~20	0
DTO1	凍防制御 基準落下温度	℃	1	1	3
DTO2	凍防制御 無変化温度偏差	deg	1	1	2
DTO3	凍防制御 ｱｯﾌﾟ温度偏差	deg	1	1	4
DTO4	凍防制御 復帰温度	deg	1	1	13
DTKTR0	凍防制御 停止温度(けつろ、再熱除湿用)	deg	0.5	-10~20	0
DTKTR1	凍防制御 基準落下温度(けつろ、再熱除湿用)	℃	1	1	3
DTKTR2	凍防制御 無変化温度偏差(けつろ、再熱除湿用)	deg	1	1	1
DTKTR3	凍防制御 ｱｯﾌﾟ温度偏差(けつろ、再熱除湿用)	deg	1	1	2
DTKTR4	凍防制御 復帰温度(けつろ、再熱除湿用)	deg	1	1	6

5。 定数決定のポイント

・TTOFGは凍防停止温度付近の温度検出ﾊﾞﾗﾂｷや、ｼｽﾃﾑの過渡影響回避を出来れば良い程度なので
3分程度が基本。TTO+TTOFGで氷結回避できれば良い。
・室外制御で、下限Hz制限をｱｯﾌﾟさせる制御があったときに、室外機は凍防よりも下限Hz制限を優先するので、凍防停止することもある。

6。 多機能との関連

室外機の凍結防止制御

7。 機能確認時にモニタしたい情報

凍防ステータス、各部ｻｰﾐｽﾀ温度、運転Hz

2006年3月6日 [redacted]

Formalisation

Following tasks, for instance, are needed during the process of formalisation of the specification.

- Clarification of variables and range of values.

Variable	Range of Variable
v1 (state_compressor)	ON, OFF
v6 (state_antifreeze)	comeback, up, same, pendency, abort
v10 (state_heat-exchange-temp)	A,B,C,D,E
tto	clearstart, over

- Clarification of the relations of variables.
  - Previous v1=OFF and v1=ON then v6=comeback and tto=clearstart.
  - tto=over and v10=A then v6=comeback.
  - …

Formal model in Agda I

```
module 20091029aircond-e where
open import Logic
open import Data.Nat hiding (_<_ ; _>_ ; _+_)
open import Data.Fin hiding (_<_ ; _+_)

data ComperssorOpStateType : Set where
  ON : ComperssorOpStateType
  OFF : ComperssorOpStateType

data TemperatureType : Set where
  -32'0℃ : TemperatureType
  -32'5℃ : TemperatureType
  -- 0.5 inteval
  +95'5℃ : TemperatureType

data OpModeType : Set where
  StopMode : OpModeType
  CoolMode : OpModeType
  WarmMode : OpModeType
  DryMode : OpModeType
  ReheatDryMode : OpModeType

data ExteriorFreezePreventionType : Set where
  Normal : ExteriorFreezePreventionType
  FreezePrevention : ExteriorFreezePreventionType

data FreezePreventionStateType : Set where
  Return : FreezePreventionStateType
  Up : FreezePreventionStateType
  NoChange : FreezePreventionStateType
  Droop : FreezePreventionStateType
  Stopped : FreezePreventionStateType

RevolutionType : Set
RevolutionType = Fin 150 -- ×10 rpm

data HEXTempreatureZoneType : Set where
  A : HEXTempreatureZoneType
  B : HEXTempreatureZoneType
  C : HEXTempreatureZoneType
  D : HEXTempreatureZoneType
  E : HEXTempreatureZoneType
```

Formal model in Agda II

```
data InHEXTemperatureStateType : Set where
  GoingDown : InHEXTemperatureStateType
  GoingUp : InHEXTemperatureStateType
record State : Set where
  field
    CompressorOpState : ComperssorOpStateType --
      input
    InteriorHEXTemperature : TemperatureType -- input
    OpMode : OpModeType -- input
    ExteriorFreezePrevention :
      ExteriorFreezePreventionType -- input
    FreezePreventionState :
      FreezePreventionStateType -- output
    InteriorFanRevolutionGoalGLB : RevolutionType --
      output
    InteriorHEXTemperatureState :
      InHEXTemperatureStateType
    HEXTemperatureZone : HEXTempreatureZoneType
    tx : ℕ -- internal variable
    tto : ℕ -- internal variable
```

```
postulate
    ttofg : ℕ
    ttofgktr : ℕ
    ttofgdry : ℕ
    dto0 : TemperatureType
    dto1 : TemperatureType
    dto2 : TemperatureType
    dto3 : TemperatureType
    dto4 : TemperatureType
    dtktr0 : TemperatureType
    dtktr1 : TemperatureType
    dtktr2 : TemperatureType
    dtktr3 : TemperatureType
    dtktr4 : TemperatureType
```

Formal model in Agda III

```
v1 = State.CompressorOpState
v2 = State.InteriorHEXTemperature
v3 = State.OpMode
-- v4 = State.室温
v5 = State.ExteriorFreezePrevention
v6 = State.FreezePreventionState
-- v7 = State.InteriorFanApproxRevolution
v8 = State.InteriorFanRevolutionGoalGLB
v9 = State.InteriorHEXTemperatureState
v10 = State.HEXTemperatureZone

postulate
  W2 : RevolutionType -- constant!
  TTO : ℕ
  DC1 : TemperatureType
  DC2 : TemperatureType
  DC3 : TemperatureType
  DC4 : TemperatureType
  DC5 : TemperatureType
```

```
postulate
  pre : State -- previous state
  current : State -- current state
  inAzone : HEXTempreatureZoneType → Pr
  inBzone : HEXTempreatureZoneType → Pr
  inCzone : HEXTempreatureZoneType → Pr
  inDzone : HEXTempreatureZoneType → Pr
  inEzone : HEXTempreatureZoneType → Pr
  Counting : ℕ → Pr
  CountOver : ℕ → Pr
  Start : Pr → Pr
  ClearStart : ℕ → Pr
  Stop : ℕ → Pr
  _>=_ : TemperatureType → TemperatureType → Pr
  _<_  : TemperatureType → TemperatureType → Pr
  _>_  : TemperatureType → TemperatureType → Pr
  _+_ :  TemperatureType → TemperatureType
    → TemperatureType
```

Formal model in Agda IV

```
prop1 : Pr
prop1 = v1 pre ≡ OFF ∧ v1 current ≡ ON
      → v6 current ≡ Return ∧ ClearStart (State.tto
        current)

prop2 : Pr
prop2 =
  CountOver (State.tto current) ∧ inAzone (v10 current)
  → v6 current ≡ Return
prop3 : Pr
prop3 =
  CountOver (State.tto current) ∧ inBzone (v10 current)
  → v6 current ≡ Up
prop4 : Pr
prop4 =
  CountOver (State.tto current) ∧ inCzone (v10 current)
  → v6 current ≡ NoChange
prop5 : Pr
prop5 =
  CountOver (State.tto current) ∧ inDzone (v10 current)
  → v6 current ≡ Droop
```

```
prop6 : Pr
prop6 = CountOver (State.tto current) ∧ inEzone (v10
        current)
          ∧ Counting (State.tx current)
          → v6 current ≡ Droop
prop7 : Pr
prop7 = CountOver (State.tto current) ∧ inEzone (v10
        current)
          ∧ CountOver (State.tx current)
          → v6 current ≡ Stopped
prop7-1 : Pr
prop7-1 = Counting (State.tto current)
          → v6 current ≡ Return
prop8 : Pr
prop8 = inDzone (v10 pre)  ∧ inEzone (v10 current)
          → ClearStart (State.tx current)
prop9 : Pr
prop9 = inEzone (v10 pre) ∧ inDzone (v10 current)
          ∧ Counting (State.tx current)
          → Stop (State.tx current)
```

Formal model in Agda V

```
prop10 : Pr
prop10 = v1 pre ≡ ON ∧ v1 current ≡ OFF
      → Stop (State.tto current)
prop11 : Pr
prop11 =
  v6 pre ≡ Stopped ∧ ¬ inAzone (v10 pre)
  → v6 current ≡ Stopped
prop12 : Pr
prop12 = v6 current ≡ Stopped
           → v8 current ≡ W2
prop13 : Pr
prop13 =  v1 pre ≡ OFF ∧ v1 current ≡ ON
      → v8 current ≡ zero
prop14 : Pr
prop14 = v9 current ≡ GoingDown ∧ v2 current >= DC2
      → inAzone (v10 current)
prop15 : Pr
prop15 =
  v9 current ≡ GoingDown ∧ v2 current >= DC3
  ∧ v2 current < DC2
  → inBzone (v10 current)
prop16 : Pr
prop16 = v9 current ≡ GoingDown
           ∧ v2 current >= DC4 ∧ v2 current < DC3
           → inCzone (v10 current)
prop17 : Pr
prop17 = v9 current ≡ GoingDown
           ∧ v2 current >= DC5 ∧ v2 current < DC4
           → inDzone (v10 current)
prop18 : Pr
prop18 = v9 current ≡ GoingDown ∧ v2 current < DC5
           → inEzone (v10 current)
prop19 : Pr
prop19 = v9 current ≡ GoingUp ∧ v2 current >= DC1
           → inAzone (v10 current)
prop20 : Pr
prop20 = v9 current ≡ GoingUp
           ∧ v2 current >= DC2 ∧ v2 current < DC1
           → inBzone (v10 current)
```

Formal model in Agda VI

```
prop21 : Pr
prop21 = v9 current ≡ GoingUp
           ∧ v2 current >= DC3 ∧ v2 current < DC2
           → inCzone (v10 current)
prop22 : Pr
prop22 = v9 current ≡ GoingUp
           ∧ v2 current >= DC5 ∧ v2 current < DC3
           → inDzone (v10 current)
prop23 : Pr
prop23 = v9 current ≡ GoingUp ∧ v2 current < DC5
     → inEzone (v10 current)
prop24 : Pr
prop24 = v3 current ≡ ReheatDryMode
           → DC1 ≡ dtktr4 ∧ DC2 ≡ dtktr1 + dtktr3
              ∧ DC3 ≡ dtktr1 + dtktr2
              ∧ DC4 ≡ dtktr1 ∧ DC5 ≡ dtktr0
prop25 : Pr
prop25 = v3 current ≡ CoolMode ∨ v3 current ≡
      DryMode
           → DC1 ≡ dto4 ∧ DC2 ≡ dto1 + dto3
              ∧ DC3 ≡ dto1 + dto2
              ∧ DC4 ≡ dto1 ∧ DC5 ≡ dtktr0
prop26 : Pr
prop26 = v2 pre ≡ v2 current → v9 current ≡ v9 pre
prop27 : Pr
prop27 = v2 pre < v2 current → v9 current ≡ GoingUp

prop28 : Pr
prop28 = v2 pre > v2 current → v9 current ≡
      GoingDown
```

Test case

- Test case: A state *s* (or a sequence of states) in the state transition system *S* representing the Agda model.
- Expected output: *s'* such that transition $(s,s') \in S$.
- Ex 3.

Test suite (Inputs)

v1	v6	v10	tto
ON	*	*	clearstart
ON	comeback	A	over
ON	up	B	over
ON	same	C	over
ON	pendency	D	over
OFF	*	*	clearstart
OFF	comeback	A	over
OFF	up	B	over
OFF	same	C	over
OFF	pendency	D	over

Expected Outputs

v1	v6	v10	tto
*	*	*	clearstart
*	comeback	A	over
*	up	B	over
*	same	C	over
*	pendency	D	over

v1	v6	v10	tto
ON	comeback	*	clearstart
OFF	*	*	clearstart
OFF	comeback	A	over
OFF	up	B	over
OFF	same	C	over
OFF	pendency	D	over

Test case generation

- Test case generation from a formal specification by hand (for this case study.)

  Automatic generation is aimed at.

- Test case for a spec is a valuation of variables in the spec., which makes the required properties true.

  Some variables are *input* variables, others are *output* variables.

- # of valid test cases will be enormous

  ➔ boundary conditions to the required properties.

Benefits of our Framework

- By Formal Model-based Testing

  → Automation of test case generation,

  Improvement of quality of test cases.

  – In the previous work with Company R, we confirmed the improvement of test case quality by the model-based test generation using Agda.

- By Cluster in the Loop Simulation (CILS)

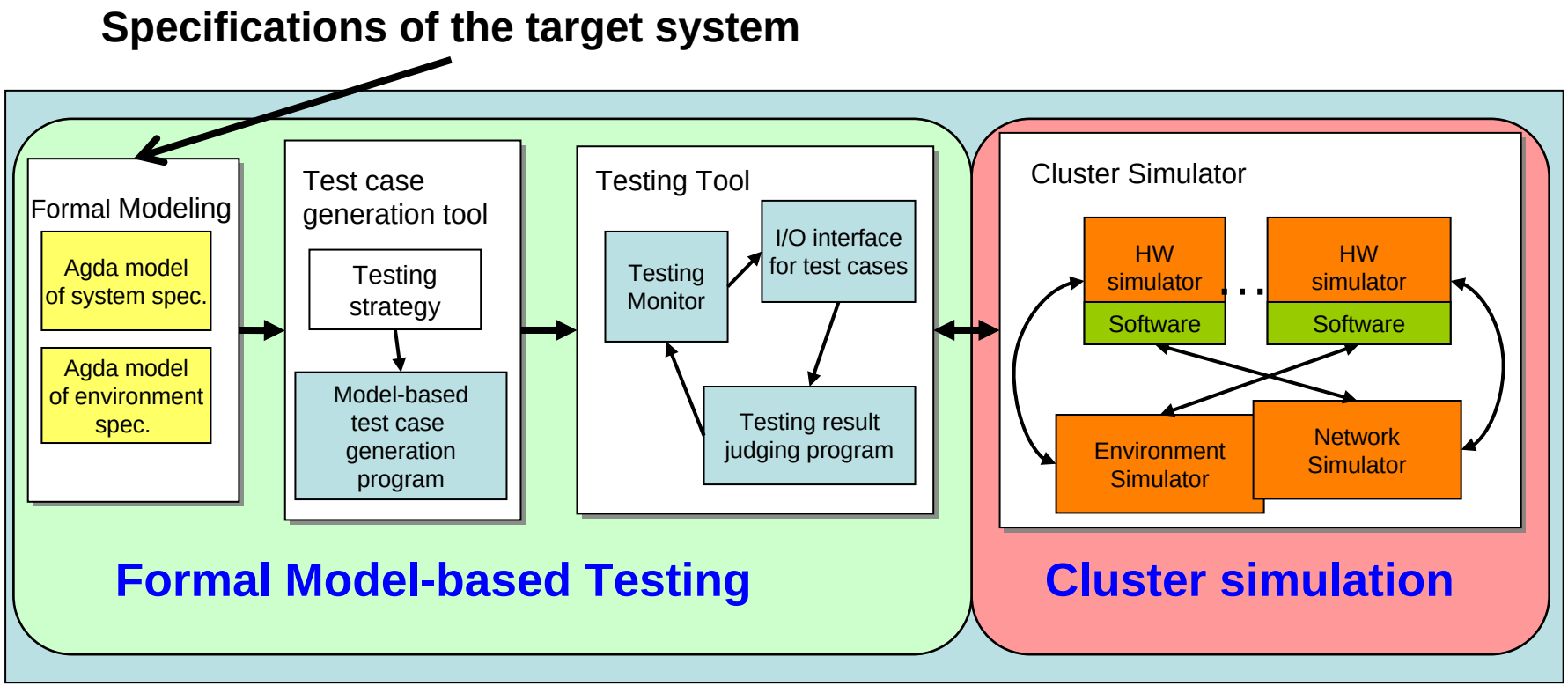
  → Automation of testing,

  Expansion of scalability,
  Reducing time for testing.

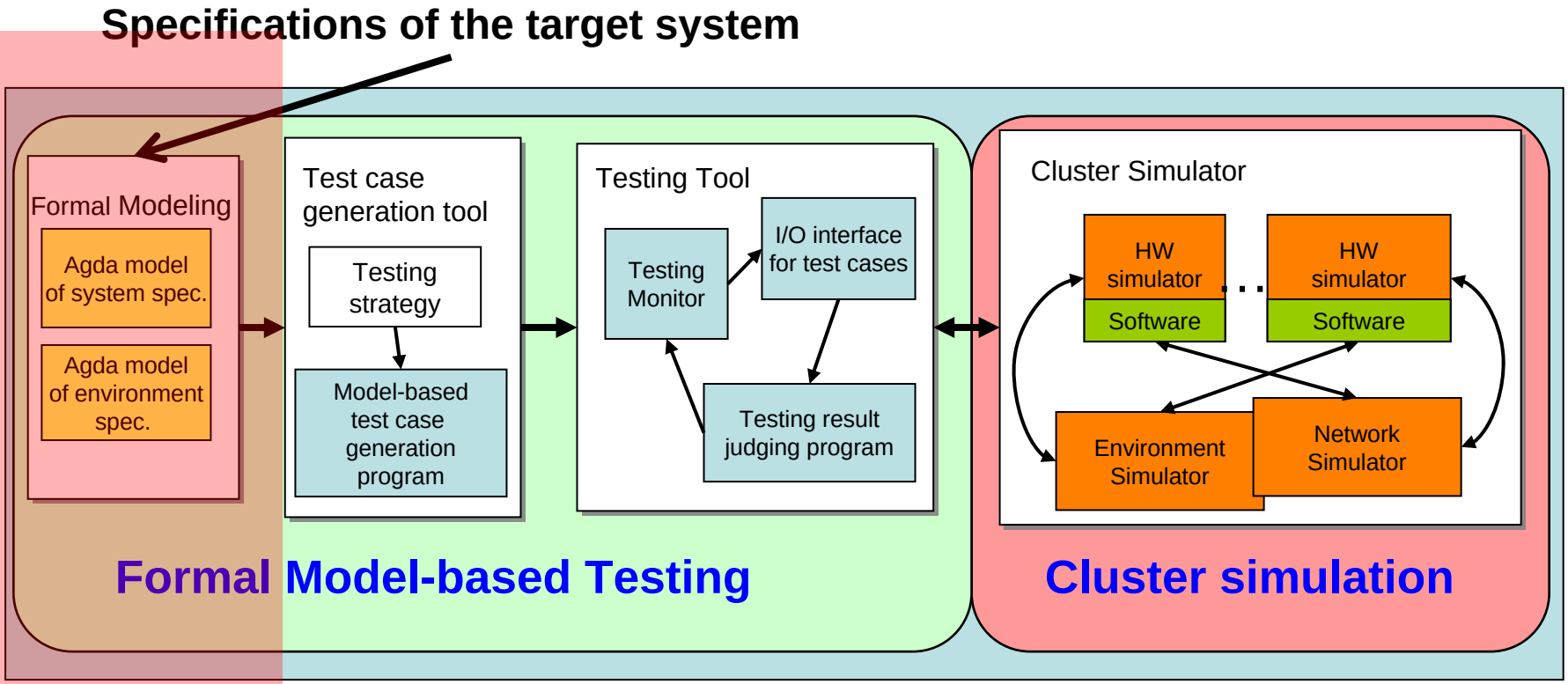
Outline

- Backgrounds
- Our testing framework
- Design of the proposed testing system

Outline Design of the proposed CILS System



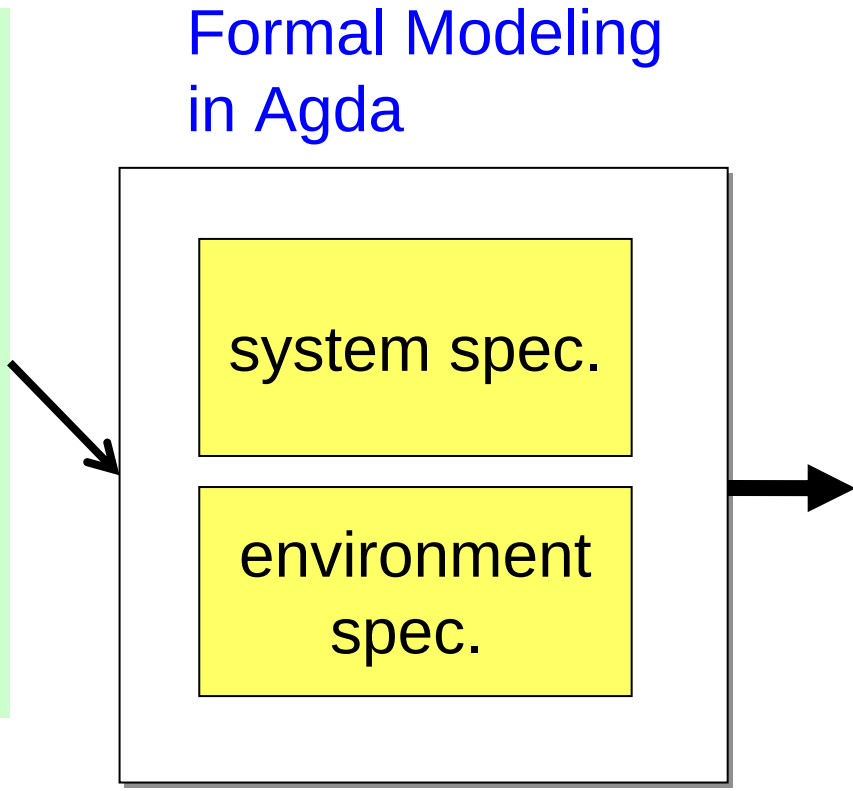
Outline Design of the proposed CILS System



Formal Modeling

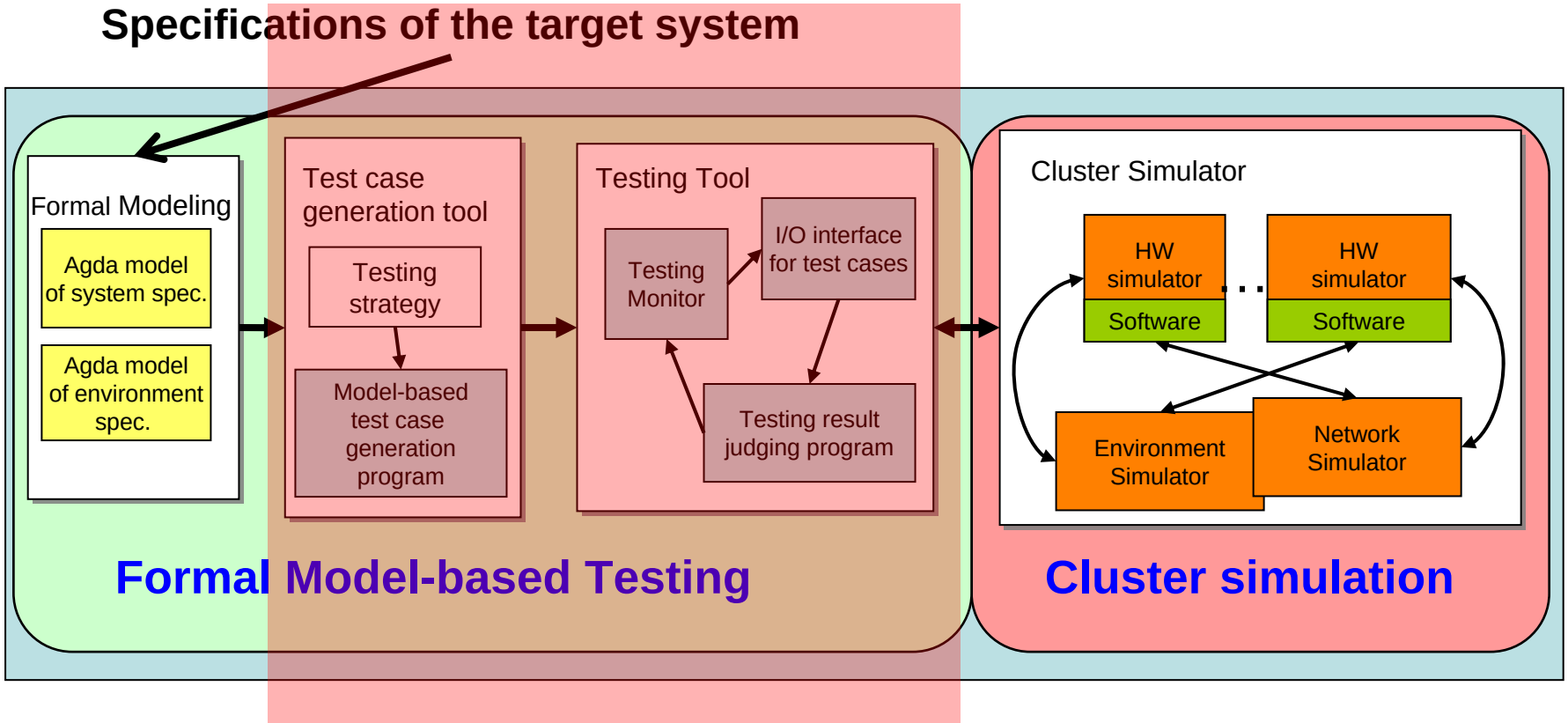
Given informal specifications

- System specification:
  - natural language text
    or state transition diagrams.
- Environment specification:
  - e.g., relations between temp. and airflow
  - domain specific knowledge.

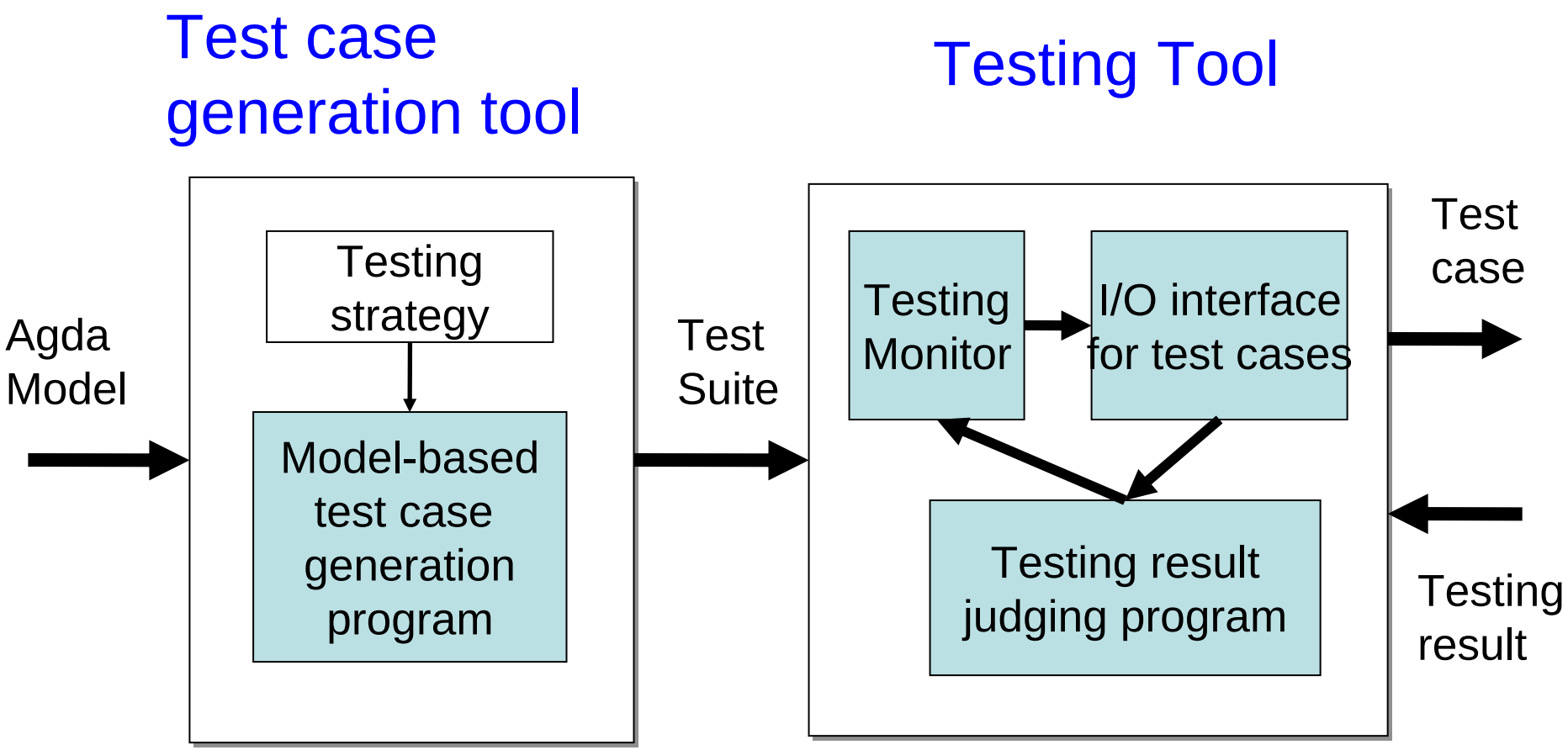


Formal Description in Agda

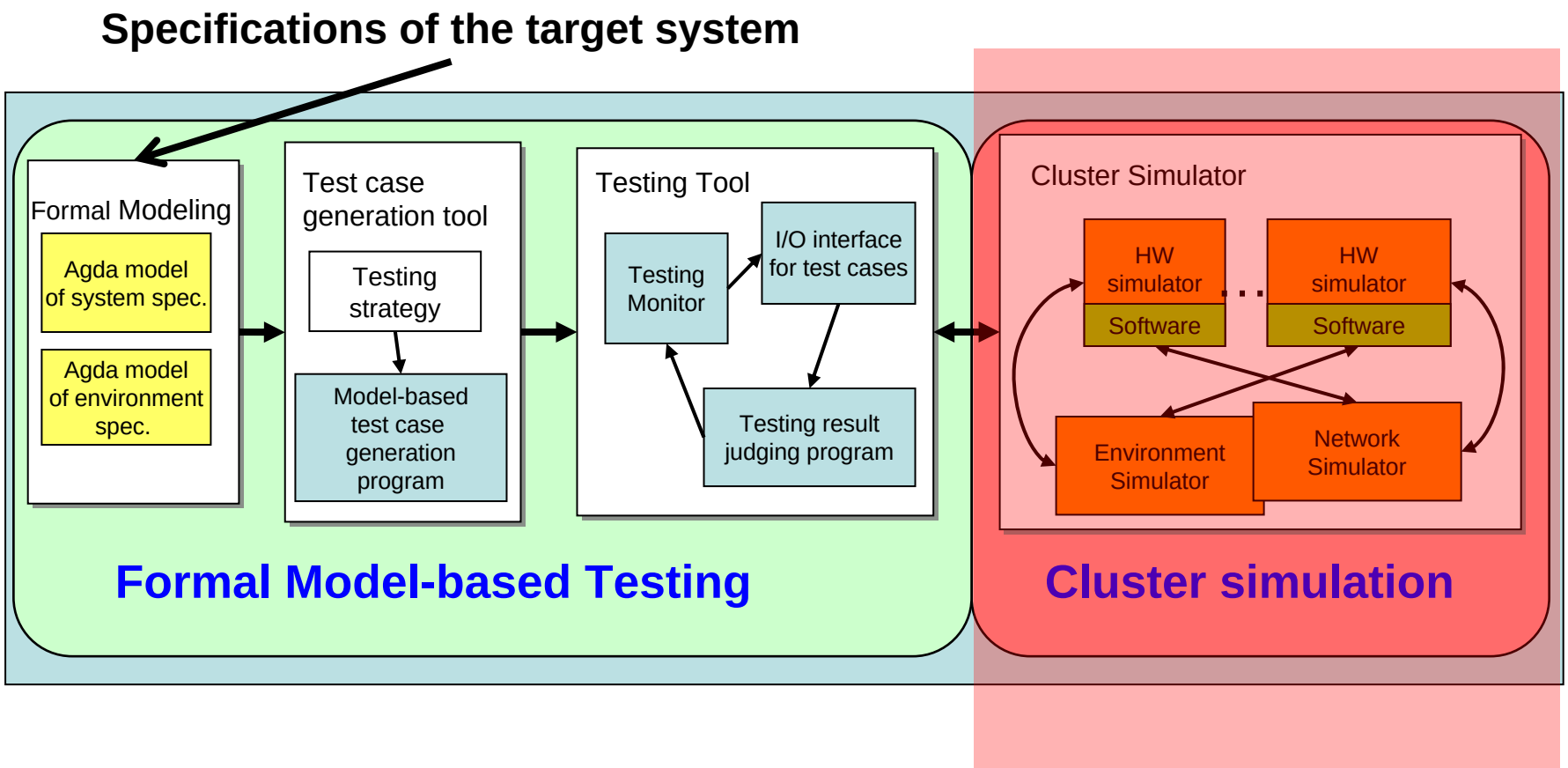
Outline Design of the proposed CILS System



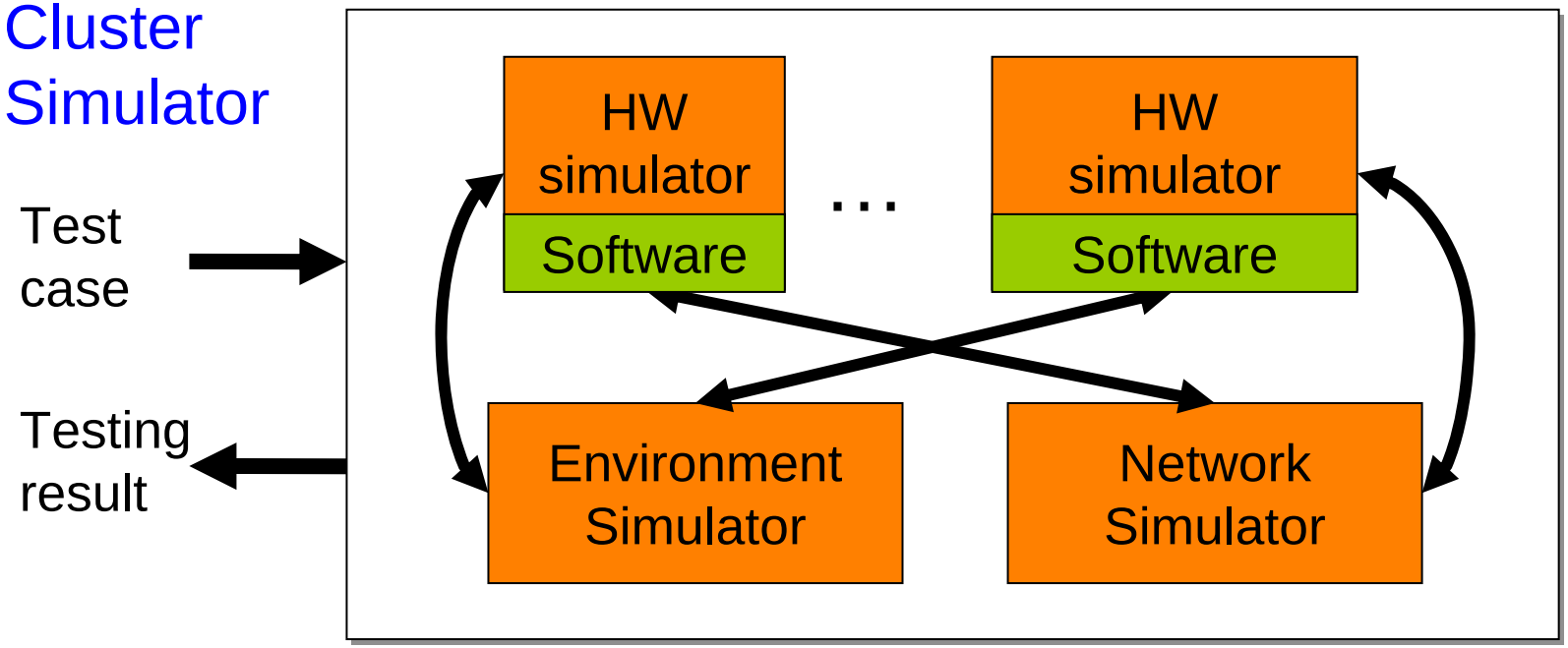
Formal Model-based Testing



Outline Design of the proposed CILS System



Cluster Simulation



- The simulators will be developed in the cluster system called SATSUKI in Collaborative Facilities for Verification in AIST.

Our Position

- Three parties got together in June 2009.
  - Feasibility study till March 2010, seeking for support.
  - Start the real project from April 2010, hopefully.
- Planned:
  - CILS based on formal-model based testing
  - Testing system for air-conditioner with automatic test case generator
  - 3 year project proposal.
- Issues to be detailed
  - Method for test case generation

    what should be boundary conditions?
  - Framework for verification (esp. need for temporal properties?)
  - Fast simulators