

TLA+/TLC: формальный метод верификации конкурентных алгоритмов для инженеров

Алексей Найдёнов

ITOOLABS 



Amazon Web Services [успешно использует TLA⁺ Tools¹](#) с 2011 года

"At AWS, formal methods have been a big success. They have helped us prevent subtle, serious bugs from reaching production, bugs that we would not have found via any other technique."

1. <http://lamport.azurewebsites.net/tla/formal-methods-amazon.pdf>

Amazon Web Services успешно использует TLA⁺ Tools¹ с 2011 года

"At AWS, formal methods have been a big success. They have helped us prevent subtle, serious bugs from reaching production, bugs that we would not have found via any other technique."

Microsoft использует TLA⁺ [во многих проектах](#)², включая Azure CosmosDB³

"TLA⁺ is not yet a prerequisite for our hiring. However, a candidate's knowledge of TLA⁺ is given significant weight in our evaluation. To us, it is a great indicator of those who really care about quality and correctness."

1. <http://lamport.azurewebsites.net/tla/formal-methods-amazon.pdf>
2. <https://www.microsoft.com/en-us/research/search/?q=TLA%2B>
3. <https://docs.microsoft.com/en-us/azure/cosmos-db/consistency-levels>

MongoDB для поиска¹ ошибок в протоколе² репликации

“We expect that formally modeling our system upfront could have saved 100s of hours of engineering time.”

1. conf.tlapl.us/07_-_TLAConf19_-_William_Schultz_-_Fixing_a_MongoDB_Replication_Protocol_Bug_with_TLA.pdf
2. github.com/will62794/mongo-repl-tla-models

MongoDB для поиска¹ ошибок в протоколе² репликации

"We expect that formally modeling our system upfront could have saved 100s of hours of engineering time."

CockroachDB для спецификации³ протокола распределенных транзакций⁴

"We found that the process of writing this specification gave us more confidence in the Parallel Commit protocol itself and in its integration into CockroachDB."

1. conf.tlapl.us/07_-_TLAConf19_-_William_Schultz_-_Fixing_a_MongoDB_Replication_Protocol_Bug_with_TLA.pdf
2. github.com/will62794/mongo-repl-tla-models
3. www.cockroachlabs.com/blog/parallel-commits
4. github.com/cockroachdb/cockroach/tree/master/docs/tla-plus

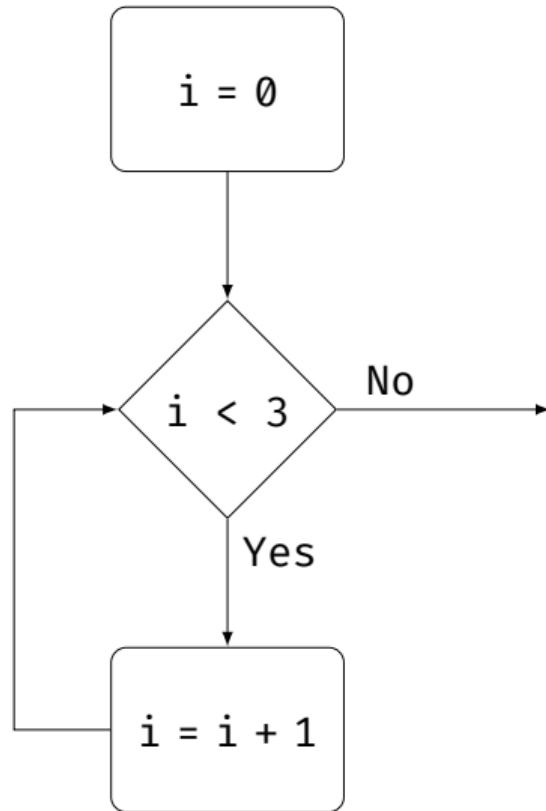


©The Cartoon Network

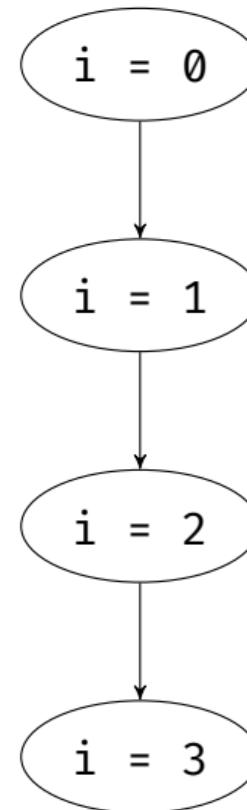
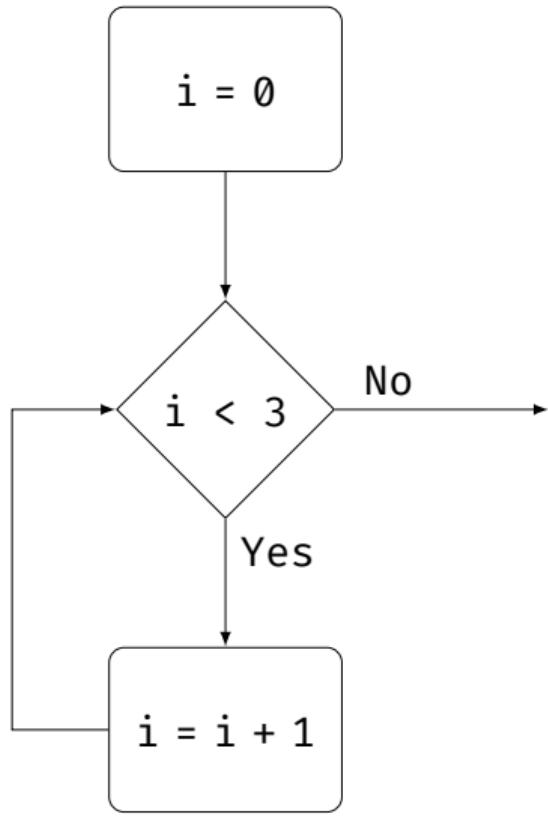
Что такое программа?

```
1 | var i = 0
2 | for i < 3 {
3 |   i = i + 1
4 | }
```

```
1 | var i = 0  
2 | for i < 3 {  
3 |   i = i + 1  
4 | }
```



```
1 | var i = 0
2 | for i < 3 {
3 |   i = i + 1
4 | }
```



a = 0
b = 0

Process 1

1 | a = b + 1
2 | b = a + 1

Process 2

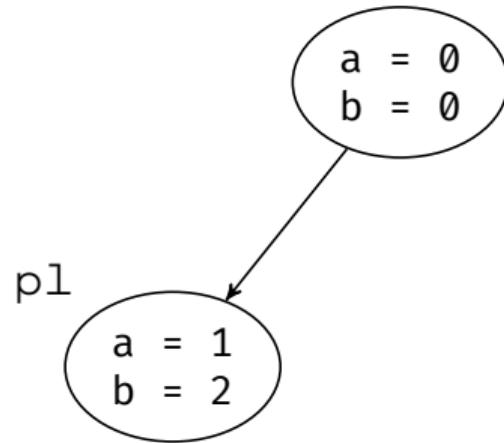
1 | b = a + 1
2 | a = b + 1

Process 1

```
1 | a = b + 1  
2 | b = a + 1
```

Process 2

```
1 | b = a + 1  
2 | a = b + 1
```

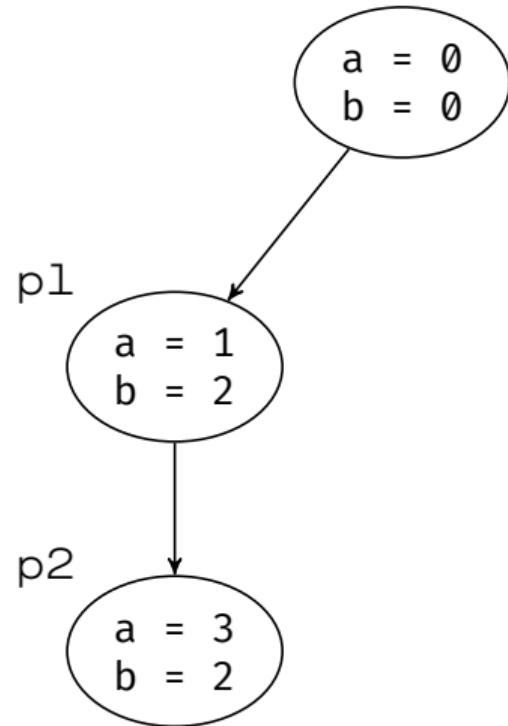


Process 1

```
1 | a = b + 1  
2 | b = a + 1
```

Process 2

```
1 | b = a + 1  
2 | a = b + 1
```

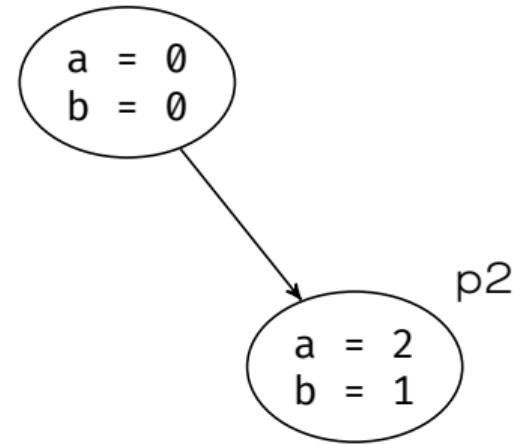


Process 1

```
1 | a = b + 1  
2 | b = a + 1
```

Process 2

```
1 | b = a + 1  
2 | a = b + 1
```

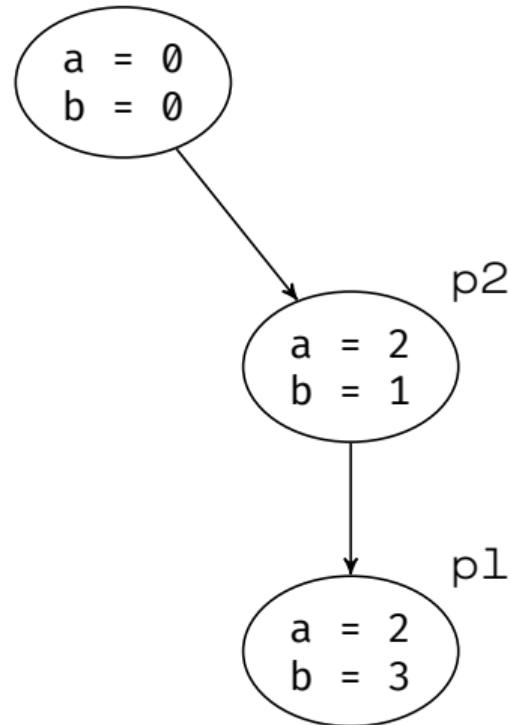


Process 1

```
1 | a = b + 1  
2 | b = a + 1
```

Process 2

```
1 | b = a + 1  
2 | a = b + 1
```

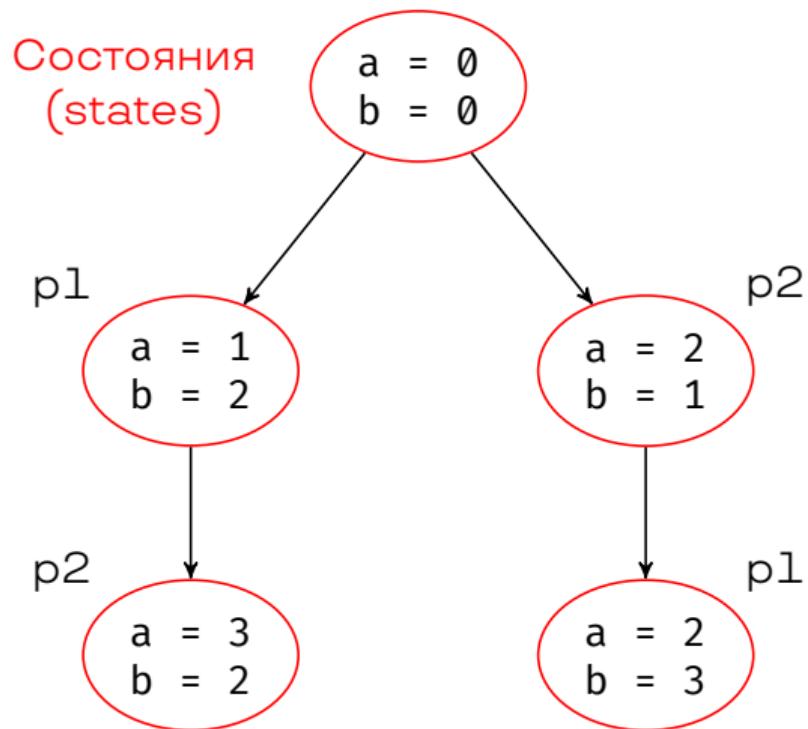


Process 1

```
1 | a = b + 1  
2 | b = a + 1
```

Process 2

```
1 | b = a + 1  
2 | a = b + 1
```

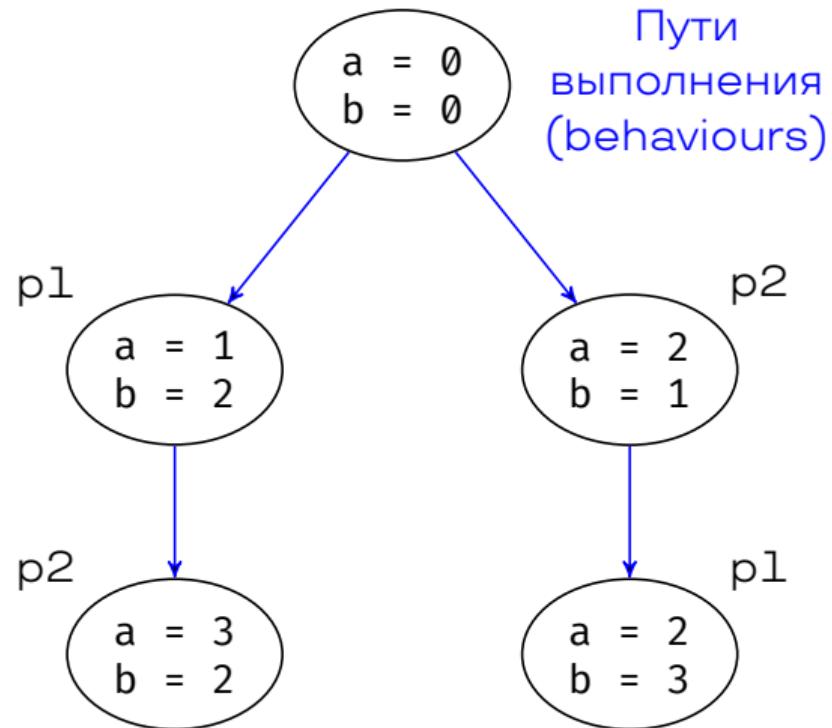


Process 1

```
1 | a = b + 1  
2 | b = a + 1
```

Process 2

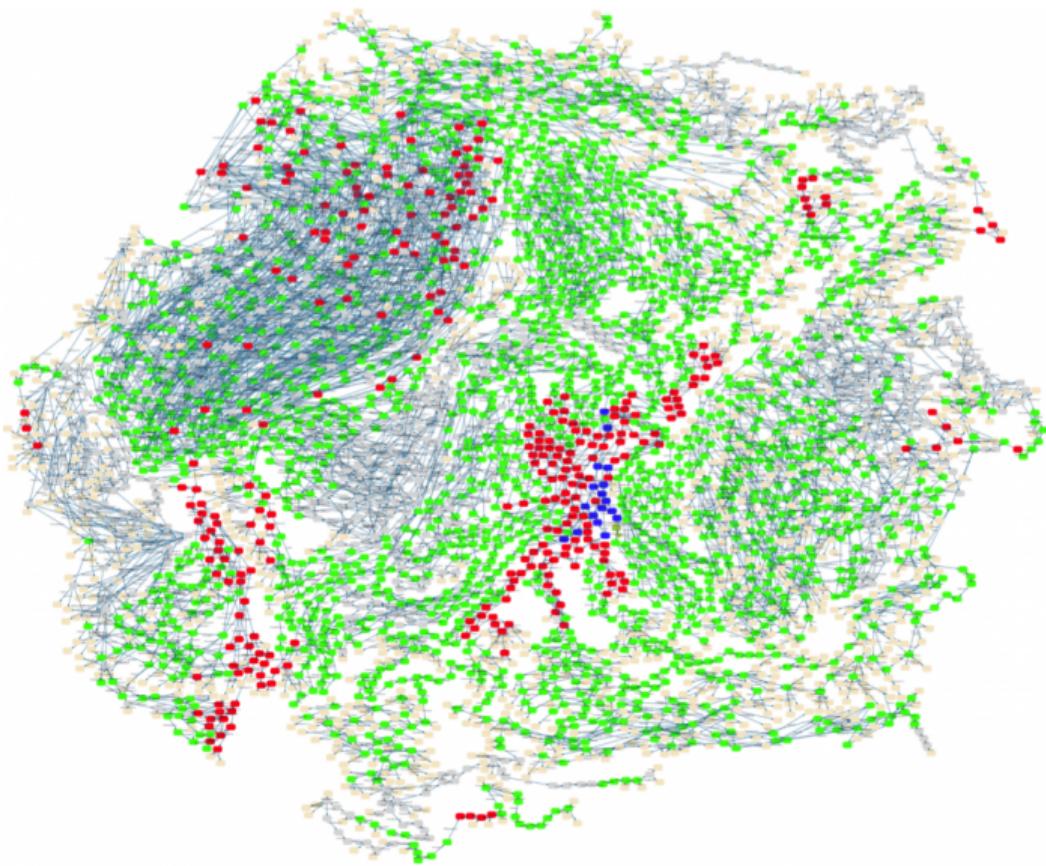
```
1 | b = a + 1  
2 | a = b + 1
```



$$(m \cdot n) \frac{(m \cdot n)!}{(n!)^m},$$

где

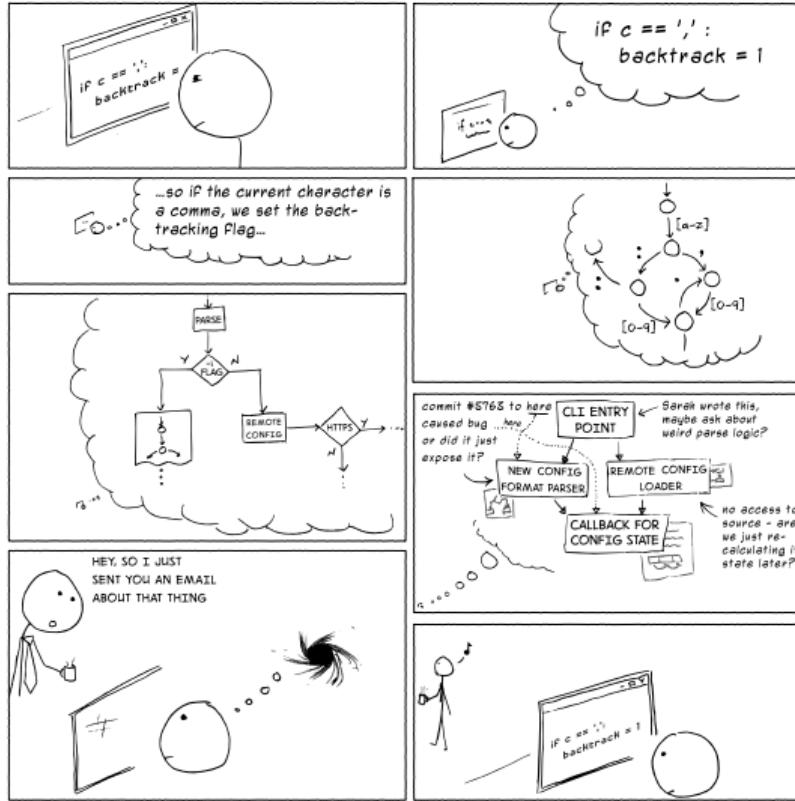
m – количество процессов
 n – длина пути выполнения



CAN Bus state space visualization

https://www3.hhu.de/stups/prob/index.php/State_space_visualization_examples

THIS IS WHY YOU SHOULDN'T INTERRUPT A PROGRAMMER



© Jason Heeris 2013

LICENSE: CC BY-NC-ND 2.5 AU

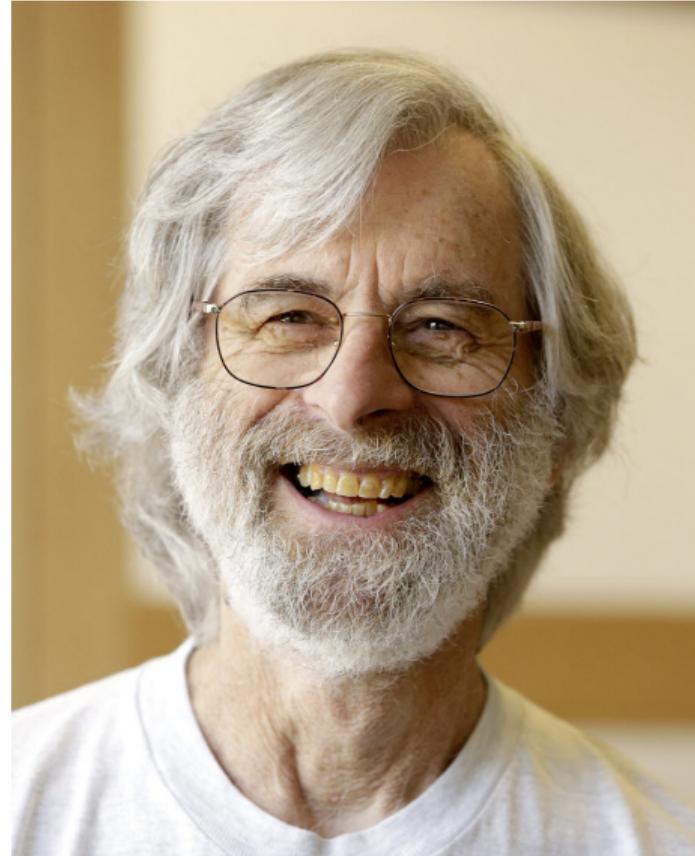
heeris.id.au

“A distributed system is one in which the failure of a computer you didn’t even know existed can render your own computer unusable.”

— Leslie Lamport, May 1987

Лесли Лэмпорт (Leslie Lamport)

- Lamport timestamps
- Bakery algorithm
- PAXOS
- LaTeX
- **TLA⁺**



Temporal Logic of Actions

Математическая логика

Математическая логика

$A \wedge B$

Conjunction

Математическая логика

$A \wedge B$

Conjunction

$A \vee B$

Disjunction

Математическая логика

$A \wedge B$

Conjunction

$A \vee B$

Disjunction

$\neg A$

Negation

Математическая логика

$A \wedge B$

Conjunction

$A \vee B$

Disjunction

$\neg A$

Negation

Темпоральная логика



©Universal Pictures Amblin Entertainment

Математическая логика

$A \wedge B$

Conjunction

$A \vee B$

Disjunction

$\neg A$

Negation

Темпоральная логика

$\Box A$

Always

Математическая логика

$A \wedge B$

Conjunction

$A \vee B$

Disjunction

$\neg A$

Negation

Темпоральная логика

$\Box A$

Always

$\neg \Box \neg A$

Математическая логика

$A \wedge B$

Conjunction

$A \vee B$

Disjunction

$\neg A$

Negation

Темпоральная логика

$\Box A$

Always

$\Diamond A$

Eventually

Математическая логика

$A \wedge B$

Conjunction

$A \vee B$

Disjunction

$\neg A$

Negation

Темпоральная логика

$\Box A$

Always

$\Diamond A$

Eventually

$\Diamond \Box A$

Математическая логика

$A \wedge B$

Conjunction

$A \vee B$

Disjunction

$\neg A$

Negation

Темпоральная логика

$\Box A$

Always

$\Diamond A$

Eventually

$\Diamond \Box A$

Eventually always

Математическая логика

$A \wedge B$

Conjunction

$A \vee B$

Disjunction

$\neg A$

Negation

Темпоральная логика

$\Box A$

Always

$\Diamond A$

Eventually

$\Diamond \Box A$

Eventually always

p'

Математическая логика

$A \wedge B$

Conjunction

$A \vee B$

Disjunction

$\neg A$

Negation

Темпоральная логика

$\Box A$

Always

$\Diamond A$

Eventually

$\Diamond \Box A$

Eventually always

p'

Next state

Математическая логика

$A \wedge B$	$A \setminus B$	Conjunction
$A \vee B$	$A \backslash B$	Disjunction
$\neg A$	$\sim A$	Negation

Темпоральная логика

$\Box A$	$[]A$	Always
$\Diamond A$	$<>A$	Eventually
$\Diamond \Box A$	$<>[]A$	Eventually always
p'	p'	Next state

TLA+

TRUE

TRUE

True

TLA+

TRUE
FALSE

TRUE
FALSE

True
False

TLA+

TRUE

FALSE

$A \triangleq B$

TRUE

FALSE

$A == B$

True

False

Define

TLA+

TRUE	TRUE	True
FALSE	FALSE	False
$A \triangleq B$	$A == B$	Define
$A = B$	$A = B$	Equality

TLA+

TRUE	TRUE	True
FALSE	FALSE	False
$A \triangleq B$	$A == B$	Define
$A = B$	$A = B$	Equality
$A \neq B$	$A /= B$	Inequality

TLA+

TRUE	TRUE	True
FALSE	FALSE	False
$A \triangleq B$	$A == B$	Define
$A = B$	$A = B$	Equality
$A \neq B$	$A /= B$	Inequality
$x \in S$	$x \backslash\text{in } S$	x is in S

TLA+

TRUE	TRUE	True
FALSE	FALSE	False
$A \triangleq B$	$A == B$	Define
$A = B$	$A = B$	Equality
$A \neq B$	$A /= B$	Inequality
$x \in S$	$x \text{ \in } S$	x is in S
$x \notin S$	$x \text{ \not\in } S$	x is not in S

TLA+

TRUE	TRUE	True
FALSE	FALSE	False
$A \triangleq B$	$A == B$	Define
$A = B$	$A = B$	Equality
$A \neq B$	$A /= B$	Inequality
$x \in S$	$x \backslash\text{in } S$	x is in S
$x \notin S$	$x \backslash\text{notin } S$	x is not in S
$\forall x \in S : P$	$\backslash\text{A } x \backslash\text{in } S : P$	Universal quantifier

TLA+

TRUE	TRUE	True
FALSE	FALSE	False
$A \triangleq B$	$A == B$	Define
$A = B$	$A = B$	Equality
$A \neq B$	$A /= B$	Inequality
$x \in S$	$x \backslash\text{in } S$	x is in S
$x \notin S$	$x \backslash\text{notin } S$	x is not in S
$\forall x \in S : P$	$\backslash\text{A } x \backslash\text{in } S : P$	Universal quantifier
$\exists x \in S : P$	$\backslash\text{E } x \backslash\text{in } S : P$	Existential quantifier

MODULE 00_Counter

```
EXTENDS Naturals
CONSTANTS MinValue, MaxValue
ASSUME MinValue < MaxValue
VARIABLE counter
```

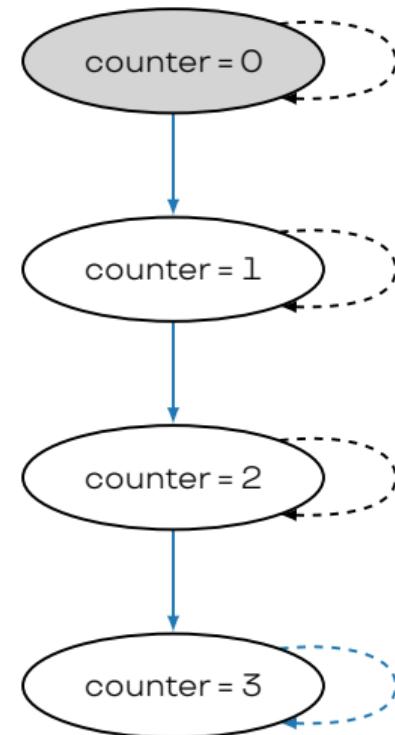
Invariant \triangleq counter \in MinValue..MaxValue

Success $\triangleq \Diamond \Box(\text{counter} = \text{MaxValue})$

Init \triangleq counter = MinValue

Next \triangleq counter' = IF counter < MaxValue
THEN counter + 1
ELSE counter

Spec \triangleq \wedge Init
 $\wedge \Box[\text{Next}]_{\text{counter}}$
 $\wedge \text{WF}_{\text{counter}}(\text{Next})$



MODULE 00_Counter

EXTENDS Naturals

CONSTANTS MinValue, MaxValue

ASSUME MinValue < MaxValue

VARIABLE counter

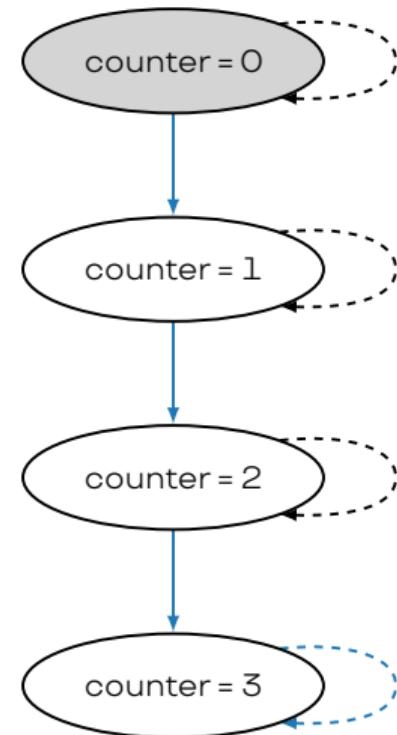
Invariant \triangleq counter \in MinValue..MaxValue

Success $\triangleq \Diamond \Box(\text{counter} = \text{MaxValue})$

Init \triangleq counter = MinValue

Next \triangleq counter' = IF counter < MaxValue
THEN counter + 1
ELSE counter

Spec \triangleq \wedge Init
 $\wedge \Box[\text{Next}]_{\text{counter}}$
 $\wedge \text{WF}_{\text{counter}}(\text{Next})$



MODULE 00_Counter

EXTENDS Naturals

CONSTANTS MinValue, MaxValue

ASSUME MinValue < MaxValue

VARIABLE counter

Invariant \triangleq counter \in MinValue..MaxValue

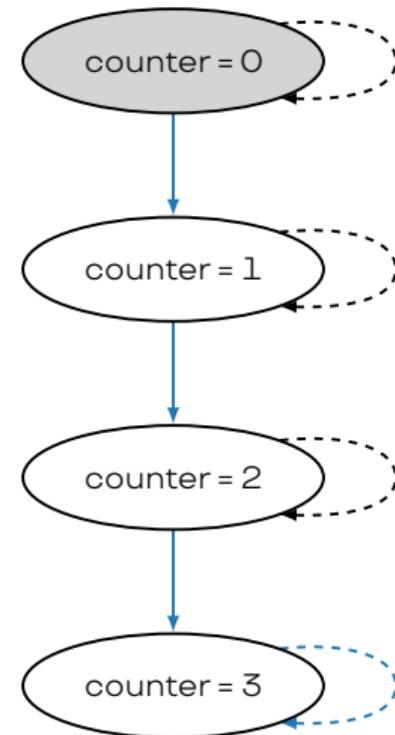
Success $\triangleq \Diamond \Box(\text{counter} = \text{MaxValue})$

Init \triangleq counter = MinValue

Next \triangleq counter' = IF counter < MaxValue
THEN counter + 1
ELSE counter

Spec \triangleq \wedge Init

$\wedge \Box[\text{Next}]_{\text{counter}}$
 $\wedge \text{WF}_{\text{counter}}(\text{Next})$



MODULE 00_Counter

```
EXTENDS Naturals
CONSTANTS MinValue, MaxValue
ASSUME MinValue < MaxValue
VARIABLE counter
```

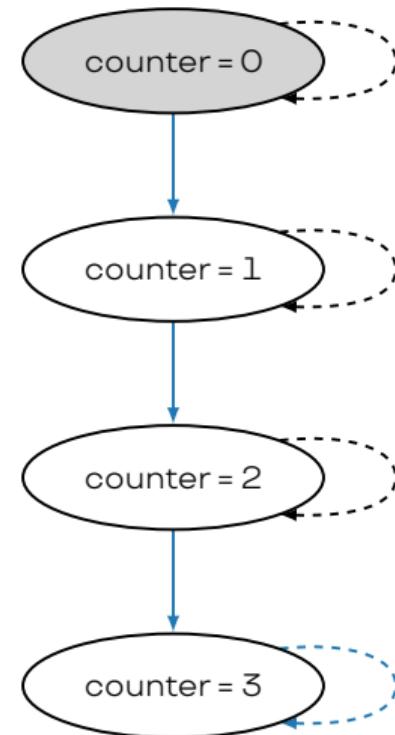
Invariant \triangleq counter \in MinValue..MaxValue

Success $\triangleq \Diamond \Box(\text{counter} = \text{MaxValue})$

Init \triangleq counter = MinValue

Next \triangleq counter' = IF counter < MaxValue
THEN counter + 1
ELSE counter

Spec \triangleq Init
 $\wedge \Box[\text{Next}]_{\text{counter}}$
 $\wedge \text{WF}_{\text{counter}}(\text{Next})$



MODULE 00_Counter

```
EXTENDS Naturals
CONSTANTS MinValue, MaxValue
ASSUME MinValue < MaxValue
VARIABLE counter
```

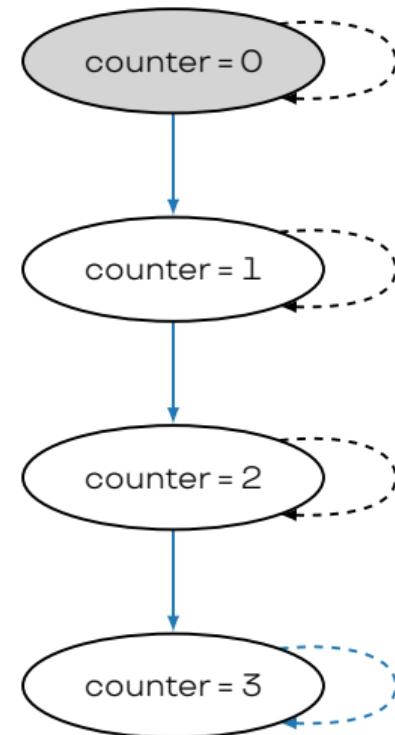
Invariant \triangleq counter \in MinValue..MaxValue

Success $\triangleq \Diamond \Box(\text{counter} = \text{MaxValue})$

Init \triangleq counter = MinValue

Next \triangleq counter' = IF counter < MaxValue
THEN counter + 1
ELSE counter

Spec \triangleq \wedge Init
 $\wedge \Box[\text{Next}]_{\text{counter}}$
 $\wedge \text{WF}_{\text{counter}}(\text{Next})$



MODULE 00_Counter

```
EXTENDS Naturals
CONSTANTS MinValue, MaxValue
ASSUME MinValue < MaxValue
VARIABLE counter
```

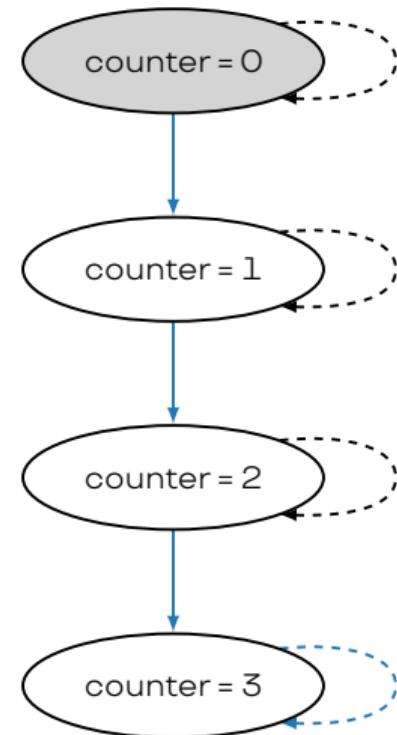
Invariant \triangleq counter \in MinValue..MaxValue

Success $\triangleq \Diamond \Box(\text{counter} = \text{MaxValue})$

Init \triangleq counter = MinValue

Next \triangleq counter' = IF counter < MaxValue
THEN counter + 1
ELSE counter

Spec \triangleq \wedge Init
 $\wedge \Box[\text{Next}]_{\text{counter}}$
 $\wedge \text{WF}_{\text{counter}}(\text{Next})$



MODULE 00_Counter

```
EXTENDS Naturals
CONSTANTS MinValue, MaxValue
ASSUME MinValue < MaxValue
VARIABLE counter
```

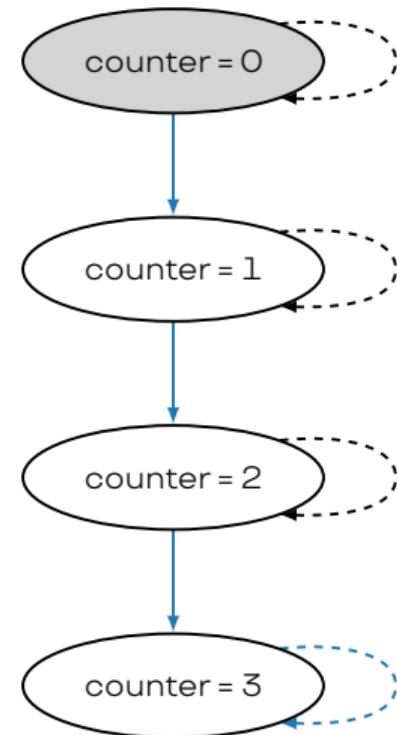
Invariant \triangleq counter \in MinValue..MaxValue

Success $\triangleq \lozenge \square (\text{counter} = \text{MaxValue})$

Init \triangleq counter = MinValue

Next \triangleq counter' = IF counter < MaxValue
THEN counter + 1
ELSE counter

Spec \triangleq \wedge Init
 $\wedge \square [\text{Next}]_{\text{counter}}$
 $\wedge \text{WF}_{\text{counter}}(\text{Next})$



MODULE 00_Counter

```
EXTENDS Naturals
CONSTANTS MinValue, MaxValue
ASSUME MinValue < MaxValue
VARIABLE counter
```

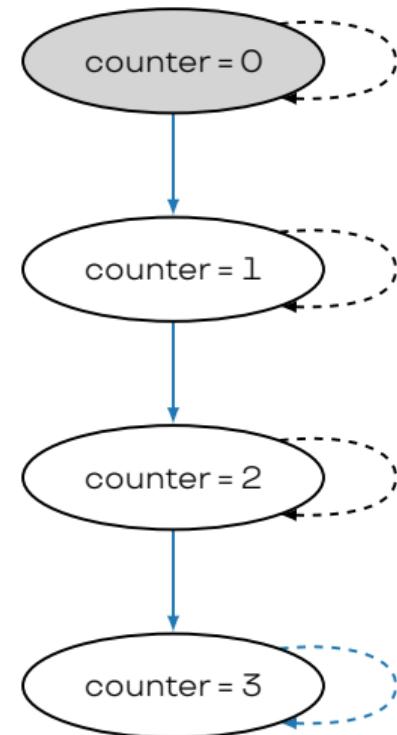
Invariant \triangleq counter \in MinValue..MaxValue

Success $\triangleq \Diamond \Box(\text{counter} = \text{MaxValue})$

Init \triangleq counter = MinValue

Next \triangleq counter' = IF counter < MaxValue
THEN counter + 1
ELSE counter

Spec \triangleq \wedge Init
 $\wedge \Box[\text{Next}]_{\text{counter}}$
 $\wedge \text{WF}_{\text{counter}}(\text{Next})$



MODULE 00_Counter

```
EXTENDS Naturals
CONSTANTS MinValue, MaxValue
ASSUME MinValue < MaxValue
VARIABLE counter
```

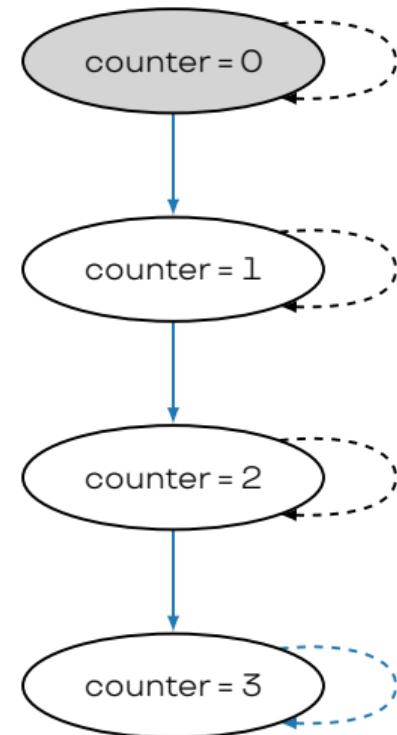
Invariant \triangleq counter \in MinValue..MaxValue

Success $\triangleq \Diamond \Box(\text{counter} = \text{MaxValue})$

Init \triangleq counter = MinValue

Next \triangleq counter' = IF counter < MaxValue
THEN counter + 1
ELSE counter

Spec \triangleq \wedge Init
 $\wedge \Box[\text{Next}]_{\text{counter}}$
 $\wedge \text{WF}_{\text{counter}}(\text{Next})$



MODULE 00_Counter

```
EXTENDS Naturals
CONSTANTS MinValue, MaxValue
ASSUME MinValue < MaxValue
VARIABLE counter
```

Invariant \triangleq counter \in MinValue..MaxValue

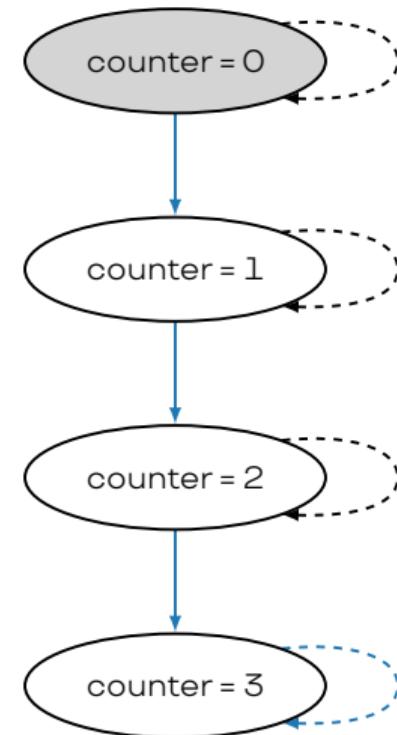
Success $\triangleq \Diamond \Box(\text{counter} = \text{MaxValue})$

Init \triangleq counter = MinValue

Next \triangleq counter' = IF counter < MaxValue
THEN counter + 1
ELSE counter

Spec \triangleq \wedge Init

$\wedge \Box[\text{Next}]_{\text{counter}}$
 $\wedge \text{WF}_{\text{counter}}(\text{Next})$



MODULE 00_Counter

```
EXTENDS Naturals
CONSTANTS MinValue, MaxValue
ASSUME MinValue < MaxValue
VARIABLE counter
```

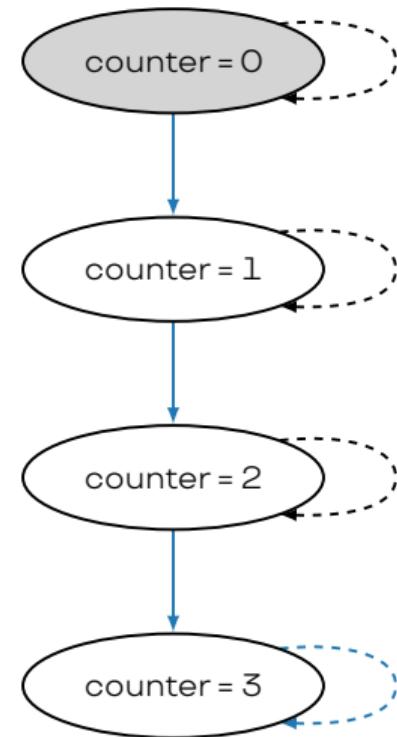
Invariant \triangleq counter \in MinValue..MaxValue

Success $\triangleq \Diamond \Box(\text{counter} = \text{MaxValue})$

Init \triangleq counter = MinValue

Next \triangleq counter' = IF counter < MaxValue
THEN counter + 1
ELSE counter

Spec $\triangleq \wedge$ Init
 $\wedge \Box[\text{Next}]_{\text{counter}}$
 $\wedge \text{WF}_{\text{counter}}(\text{Next})$



MODULE 00_Counter

```
EXTENDS Naturals
CONSTANTS MinValue, MaxValue
ASSUME MinValue < MaxValue
VARIABLE counter
```

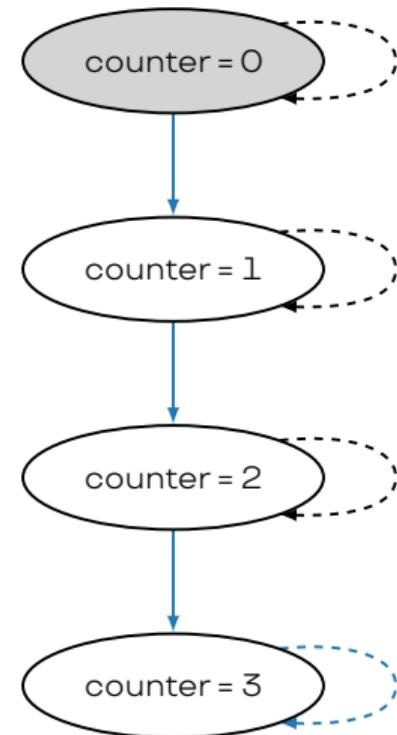
Invariant \triangleq counter \in MinValue..MaxValue

Success $\triangleq \Diamond \Box(\text{counter} = \text{MaxValue})$

Init \triangleq counter = MinValue

Next \triangleq counter' = IF counter < MaxValue
THEN counter + 1
ELSE counter

Spec \triangleq \wedge Init
 $\wedge \Box[\text{Next}]_{\text{counter}}$
 $\wedge \text{WF}_{\text{counter}}(\text{Next})$



TLC

Explicit State Model Checker for TLA⁺

TLA⁺ Toolbox by Simon Zambrovski, et al.

The screenshot shows the TLA+ Toolbox interface with several panes:

- Spec Explorer**: Shows a tree view of models: 00_Counter, Model_1, and Model_2.
- Model Overview**: For 00_Counter, it shows:
 - MODULE 00_Counter
 - EXTENDS Naturals
 - CONSTANTS MinValue, MaxValue
 - ASSUME MinValue < MaxValue
 - VARIABLE counter
 - Invariant = counter \in MinValue..MaxValue
 - Success = $\Diamond[](\text{counter} = \text{MaxValue})$
 - Init = counter = MinValue
 - Next = $\text{counter}' = \text{IF } \text{counter} < \text{MaxValue}$
THEN $\text{counter} + 1$
ELSE counter
 - Spec = $\wedge \text{Init}$
 $\wedge [][\text{Next}], \text{counter}$
 $\wedge \text{WF_counter}(\text{Next})$
- Model Overview**: For Model_1, it shows:
 - MODULE Model_1
 - EXTENDS Naturals
 - CONSTANTS MinValue, MaxValue
 - ASSUME MinValue < MaxValue
 - VARIABLE counter
 - Invariant $\triangleq \text{counter} \in \text{MinValue} .. \text{MaxValue}$
 - Success $\triangleq \Diamond\Box(\text{counter} = \text{MaxValue})$
 - Init $\triangleq \text{counter} = \text{MinValue}$
 - Next $\triangleq \text{counter}' = \text{IF } \text{counter} < \text{MaxValue}$
THEN $\text{counter} + 1$
ELSE counter
 - Spec $\triangleq \wedge \text{Init}$
 $\wedge \Box[\text{Next}], \text{counter}$
 $\wedge \text{WF_counter}(\text{Next})$
- Model Checking Results**: Shows the following details:
 - Start: 14:37:44 (Sep 9) End: 14:37:48 (Sep 9)
 - Fingerprint collision probability: calculated: 2.2E-19
 - Statistics table:

Time	Diameter	States	For Distinct	St. Queue	Siz	Module	Action	Location	States	For Distinct
00:00:04	4	5	4	0		00_Counter	Init	line 11, col 1..	1	1
00:00:04	0	1	1	1		00_Counter	Next	line 13, col 1..	4	3
 - Evaluate Constant Expression

TLA⁺ Toolbox by Simon Zambrovski, et al.

The screenshot shows the TLA+ Toolbox interface with three main panes:

- Spec Explorer** (left pane): Shows the file `00_Counter.tla` containing TLA+ code for a counter module. A red box highlights the code for the `Spec` section:

```
1 MODULE 00_Counter
2 EXTENDS Naturals
3 CONSTANTS MinValue, MaxValue
4 ASSUME MinValue < MaxValue
5 VARIABLE counter
6
7 Invariant == counter \in MinValue..MaxValue
8
9 Success == ◊[](counter = MaxValue)
10
11 Init == counter = MinValue
12
13 Next == counter' = IF counter < MaxValue
14     THEN counter + 1
15     ELSE counter
16
17 Spec == ∧ Init
18     ∧ []|Next].counter
19     ∧ WF_counter(Next)
20
21
```
- Model Overview** (bottom-left pane): Displays the `Model_1` model with tabs for `Model Overview` and `Spec Options`. It includes sections for `Model description`, `Additional Spec Options`, and fields for `Temporal formula` (with `Spec`) and `What is the model?` (with declarations for `MinValue` and `MaxValue`).
- Model Checking Results** (right pane): Shows results for `Model_1` from `Start: 14:37:44 (Sep 9)` to `End: 14:37:48 (Sep 9)`. It includes a table of state space progress and a section for `Evaluate Constant Expression`.

The bottom right corner of the interface shows the status `Spec Status : Parsed`.

TLA⁺ Toolbox by Simon Zambrovski, et al.

The screenshot shows the TLA+ Toolbox interface with several panes:

- Spec Explorer** pane: Shows a tree view of models. The root node is `00_Counter`, which has a module named `00_Counter`. This module contains code defining constants, assumptions, variables, and actions. A section of the code for action `Spec` is highlighted with a blue background.
- Model Overview** pane (highlighted with a red border): Shows the `Model_2` model. It includes sections for "What is the behavior spec?", "Temporal formula" (containing `Spec`), and "What is the model?" (specifying constant values `MinValue <- 0` and `MaxValue <- 3`).
- Model Checking Results** pane: Shows results for `Model_1`. It includes a "General" section with start and end times (14:37:44 and 14:37:48) and a "Statistics" section showing state space progress and sub-actions of next-state.
- Console** pane: Displays the TLA+ code for `00_Counter.tla`.

```
MODULE 00_Counter
EXTENDS Naturals
CONSTANTS MinValue, MaxValue
ASSUME MinValue < MaxValue
VARIABLE counter
Invariant == counter \in MinValue..MaxValue
Success == ◊[](counter = MaxValue)
Init == counter = MinValue
Next == counter' = IF counter < MaxValue
THEN counter + 1
ELSE counter
Spec == ∧ Init
∧ []|Next].counter
∧ WF_counter(Next)
```

Time	Diameter	States	Fo	Distinct	St.	Queue	Siz	Module	Action	Location	States	Fo	Distinct
00:00:04	4	5	4	0				00_Counter	Init	line 11, col 1..1	1	1	
00:00:04	0	1	1	1				00_Counter	Next	line 13, col 1..4	4	3	

TLA⁺ Toolbox by Simon Zambrovski, et al.

The screenshot shows the TLA+ Toolbox interface with several panes:

- Spec Explorer**: Shows a tree view of models: 00_Counter, Model_1, and Model_2.
- Model Overview**: For 00_Counter, it shows:
 - MODULE 00_Counter
 - EXTENDS Naturals
 - CONSTANTS MinValue, MaxValue
 - ASSUME MinValue < MaxValue
 - VARIABLE counter
 - Invariant = $\text{counter} \in \text{MinValue}..\text{MaxValue}$
 - Success = $\Diamond[](\text{counter} = \text{MaxValue})$
 - Init = $\text{counter} = \text{MinValue}$
 - Next = $\text{counter}' = \text{IF } \text{counter} < \text{MaxValue} \text{ THEN } \text{counter} + 1 \text{ ELSE } \text{counter}$
 - Spec = $\wedge \text{Init} \wedge [][\text{Next}], \text{counter} \wedge \text{WF_counter}(\text{Next})$
- Model Checking Results**: For Model_1, it shows:
 - Start: 14:37:44 (Sep 9) End: 14:37:48 (Sep 9)
 - Fingerprint collision probability: calculated: 2.2E-19
 - Statistics table (at 00:00:04):

Time	Diameter	States	For Distinct	St. Queue	Siz	Module	Action	Location	States	For Distinct
00:00:04	4	5	4	0		00_Counter	Init	line 11, col 1..1	1	1
00:00:04	0	1	1	1		00_Counter	Next	line 13, col 1..4	4	3
- Console**: Shows the command-line interface for the TLA+ specification.
- Help**: Provides help documentation for the TLA+ Toolbox.
- Spec Status**: Shows "Parsed" status.

VS Code / TLA+ Support by Андрей Лыгин @alygin

The screenshot shows the Visual Studio Code interface with several panes:

- EXPLORER**: Shows a project structure with files: .vscode, states, 00_Counter.cfg, and 00_Counter.tla.
- 00_Counter.tla**: The main editor pane containing TLA+ code for a counter module. The code defines a module 00_Counter that extends Naturals, has constants MinValue and MaxValue, assumes MinValue < MaxValue, and has a variable counter. It includes an invariant that counter is between MinValue and MaxValue, and a success condition where counter equals MaxValue. It also defines an init state where counter is MinValue, and a next state where counter is either counter + 1 if counter is less than MaxValue, or counter otherwise. A specification section at the bottom includes an invariant that Next is a counter, and a well-formedness condition for Next.
- TLA+ model checking**: A status pane showing the results of a model check. It indicates a success with a fingerprint collision probability of 2.2E-19, starting and ending at 16:00:28 on Sep 9. It also displays a table of states with columns: Time, Diameter, Found, Distinct, and Queue, with values: 00:00:00, 4, 5, 4, 0.
- 00_Counter.cfg**: Another editor pane showing the configuration file 00_Counter.cfg. It specifies a specification named Spec, constants MinValue = 0 and MaxValue = 3, and properties Invariant and Success.

VS Code / TLA+ Support by Андрей Лыгин @alygin

The screenshot shows the Visual Studio Code interface with several windows open, demonstrating TLA+ support.

- EXPLORER:** Shows the project structure with files: .vscode, states, 00_Counter.cfg, and 00_Counter.tla.
- 00_Counter.tla — example:** The main code editor window containing the TLA+ specification for a counter. It includes:
 - MODULE 00_Counter**
 - EXTENDS Naturals**
 - CONSTANTS MinValue, MaxValue**
 - ASSUME MinValue < MaxValue**
 - VARIABLE counter**
 - Invariant = counter \in minValue..MaxValue**
 - Success = <>[](counter = MaxValue)**
 - Init = counter = MinValue**
 - Next = counter' = IF counter < MaxValue THEN counter + 1 ELSE counter**
 - Spec = □ Init ∧ []□[Next]_counter ∧ WF_counter(Next)**
- TLA+ model checking — Status:** A panel showing the status of the model checking process.
 - Checking 00_Counter.tla / 00_Counter.cfg
 - Success: Fingerprint collision probability: 2.2E-19
 - Start: 16:00:28 (Sep 9), end: 16:00:28 (Sep 9)
- States:** A table showing the state metrics.

Time	Diameter	Found	Distinct	Queue
00:00:00	4	5	4	0
- 00_Counter.cfg — Specification Spec:** The configuration file containing:
 - SPECIFICATION Spec**
 - CONSTANTS**
 - MinValue = 0**
 - MaxValue = 3**
 - INVARIANTS**
 - Invariant**
 - PROPERTIES**
 - Success**

VS Code / TLA+ Support by Андрей Лыгин @alygin

The screenshot shows the Visual Studio Code interface with the TLA+ extension installed. The left sidebar displays the project structure under 'EXAMPLE' with files like '.vscode', 'states', '00_Counter.cfg', and '00_Counter.tla'. The main editor area shows the TLA+ module '00_Counter.tla' containing code for a counter that increments from minValue to maxValue. The status bar at the bottom indicates the file is 00_Counter.tla — example, with 19 lines of code, and the bottom right shows file statistics: Ln 1, Col 1, Spaces: 4, UTF-8, LF, TLA+, etc.

00_Counter.tla — example

```
00_Counter.tla x
00_Counter.tla > {} 00_Counter
1 |----- MODULE 00_Counter -----
2 EXTENDS Naturals
3 CONSTANTS MinValue, MaxValue
4 ASSUME MinValue < MaxValue
5 VARIABLE counter
6
7 Invariant = counter \in minValue..MaxValue
8
9 Success = <>[](counter = MaxValue)
10
11 Init = counter = minValue
12
13 Next = counter' = IF counter < MaxValue
14 | THEN counter + 1
15 | ELSE counter
16
17 Spec = & Init
18 | & [][]{Next}_counter
19 | & WF_counter(Next)
20
21 =====
22
```

TLA+ model checking x

Status

Checking 00_Counter.tla / 00_Counter.cfg

Success: Fingerprint collision probability: 2.2E-19

Start: 16:00:28 (Sep 9), end: 16:00:28 (Sep 9)

States

Time	Diameter	Found	Distinct	Queue
00:00:00	4	5	4	0

00_Counter.cfg x

00_Counter.cfg

```
1 SPECIFICATION Spec
2
3 CONSTANTS
4   minValue = 0
5   maxValue = 3
6
7 INVARIANTS
8   Invariant
9
10 PROPERTIES
11   Success
```

VS Code / TLA+ Support by Андрей Лыгин @alygin

The screenshot shows the VS Code interface with several open files and a TLA+ model checking window.

Left Sidebar: Shows the Explorer, Open Editors, and Project Components sections. The file `00_Counter.tla` is selected in the Open Editors section.

00_Counter.tla (Editor):

```
00_Counter.tla — example
00_Counter.tla > {} 00_Counter
  MODULE 00_Counter
  EXTENDS Naturals
  CONSTANTS MinValue, MaxValue
  ASSUME MinValue < MaxValue
  VARIABLE counter
  Invariant = counter \in MinValue..MaxValue
  Success = <>[](counter = MaxValue)
  Init = counter = MinValue
  Next = counter' = IF counter < MaxValue
    THEN counter + 1
    ELSE counter
  Spec = & Init
    & [][]{Next}_counter
    & WF_counter(Next)
=====
22
```

TLA+ model checking (Panel):

- Status:** Checking `00_Counter.tla / 00_Counter.cfg`. Success: Fingerprint collision probability: 2.2E-19. Start: 16:00:28 (Sep 9), end: 16:00:28 (Sep 9).
- States:** Time 00:00:00, Diameter 4, Found 5, Distinct 4, Queue 0.

00_Counter.cfg (Editor):

```
00_Counter.cfg — example
  SPECIFICATION Spec
  CONSTANTS
    MinValue = 0
    MaxValue = 3
  INVARIANTS
    Invariant
  PROPERTIES
    Success
```

Bottom Status Bar: Shows file paths, Alexey, Live Share, TLC, and other status indicators.

PlusCal

PlusCal

```
4 (*--algorithm 01_PCDemo
5
6 variables
7   a = 0,
8   b = 0;
9
10 define
11   Success == <>[ ](a+b = 5)
12 end define;
13
14 fair process proc1 = 1
15 begin p1:
16   a := b + 1;
17   b := a + 1;
18 end process;
19
20 fair process proc2 = 2
21 begin p2:
22   b := a + 1;
23   a := b + 1;
24 end process;
25
26 end algorithm; *)
```

PlusCal

Алгоритм

```
4 (*--algorithm 01_PCDemo
5
6 variables
7   a = 0,
8   b = 0;
9
10 define
11   Success == <>[ ](a+b = 5)
12 end define;
13
14 fair process proc1 = 1
15 begin p1:
16   a := b + 1;
17   b := a + 1;
18 end process;
19
20 fair process proc2 = 2
21 begin p2:
22   b := a + 1;
23   a := b + 1;
24 end process;
25
26 end algorithm; *)
```

PlusCal

Переменные

```
4 (*--algorithm 01_PCDemo
5
6 variables
7   a = 0,
8   b = 0;
9
10 define
11   Success == <>[ ](a+b = 5)
12 end define;
13
14 fair process proc1 = 1
15 begin p1:
16   a := b + 1;
17   b := a + 1;
18 end process;
19
20 fair process proc2 = 2
21 begin p2:
22   b := a + 1;
23   a := b + 1;
24 end process;
25
26 end algorithm; *)
```

PlusCal

Определения TLA⁺

```
4 (*--algorithm 01_PCDemo
5
6 variables
7   a = 0,
8   b = 0;
9
10 define
11   Success == <>[ ](a+b = 5)
12 end define;
13
14 fair process proc1 = 1
15 begin p1:
16   a := b + 1;
17   b := a + 1;
18 end process;
19
20 fair process proc2 = 2
21 begin p2:
22   b := a + 1;
23   a := b + 1;
24 end process;
25
26 end algorithm; *)
```

PlusCal

Процессы

```
4 (*--algorithm 01_PCDemo
5
6 variables
7   a = 0,
8   b = 0;
9
10 define
11   Success == <>[ ](a+b = 5)
12 end define;
13
14 fair process proc1 = 1
15 begin p1:
16   a := b + 1;
17   b := a + 1;
18 end process;
19
20 fair process proc2 = 2
21 begin p2:
22   b := a + 1;
23   a := b + 1;
24 end process;
25
26 end algorithm; *)
```

PlusCal

Метки

```
4 (*--algorithm 01_PCDemo
5
6 variables
7   a = 0,
8   b = 0;
9
10 define
11   Success == <>[ ](a+b = 5)
12 end define;
```

```
14   fair process proc1 = 1
15     begin p1:
16       a := b + 1;
17       b := a + 1;
18     end process;
19
20   fair process proc2 = 2
21     begin p2:
22       b := a + 1;
23       a := b + 1;
24     end process;
25
26 end algorithm; *)
```

PlusCal

Трансляция

```
27 | \* BEGIN TRANSLATION
28 | VARIABLES a, b, pc
29 |
30 | (* define statement *)
31 | Success == <>[](a+b = 5)
32 | vars == << a, b, pc >>
33 |
34 | ProcSet == {1} \cup {2}
35 |
36 | Init == (* Global variables *)
37 |   /\ a = 0
38 |   /\ b = 0
39 |   /\ pc = [self \in ProcSet |-> CASE self = 1 -> "p1"
40 |             [] self = 2 -> "p2"]
41 |
42 |
```

PlusCal

Трансляция

```
27 | \* BEGIN TRANSLATION
28 | VARIABLES a, b, pc
29 |
30 | (* define statement *)
31 | Success == <>[](a+b = 5)
32 | vars == << a, b, pc >>
33 |
34 | ProcSet == {1} \cup {2}
35 |
36 | Init == (* Global variables *)
37 |   /\ a = 0
38 |   /\ b = 0
39 |   /\ pc = [self \in ProcSet |-> CASE self = 1 -> "p1"
40 |             [] self = 2 -> "p2"]
```

PlusCal

Трансляция

```
27 | \* BEGIN TRANSLATION
28 | VARIABLES a, b, pc
29 |
30 | (* define statement *)
31 | Success == <>[](a+b = 5)
32 | vars == << a, b, pc >>
33 |
34 | ProcSet == {1} \cup {2}
35 |
36 | Init == (* Global variables *)
37 |   /\ a = 0
38 |   /\ b = 0
39 |   /\ pc = [self \in ProcSet |-> CASE self = 1 -> "p1"
40 |             [] self = 2 -> "p2"]
41 |
42 |
```

PlusCal

Трансляция

```
44 | p1 == /\ pc[1] = "p1"
45 |   /\ a' = b + 1
46 |   /\ b' = a' + 1
47 |   /\ pc' = [pc EXCEPT ![1] = "Done"]
48 |
49 proc1 == p1
50
51 p2 == /\ pc[2] = "p2"
52 |   /\ b' = a + 1
53 |   /\ a' = b' + 1
54 |   /\ pc' = [pc EXCEPT ![2] = "Done"]
55 |
56 proc2 == p2
```

PlusCal

Трансляция

```
44 | p1 == /\ pc[1] = "p1"
45 |   /\ a' = b + 1
46 |   /\ b' = a' + 1
47 |   /\ pc' = [pc EXCEPT ![1] = "Done"]
48 |
49 proc1 == p1
50
51 p2 == /\ pc[2] = "p2"
52 |   /\ b' = a + 1
53 |   /\ a' = b' + 1
54 |   /\ pc' = [pc EXCEPT ![2] = "Done"]
55 |
56 proc2 == p2
```

PlusCal

Трансляция

```
58 (* Allow infinite stuttering to prevent deadlock on termination.  
   ↳ *)  
59 Terminating == /\ \A self \in ProcSet: pc[self] = "Done"  
60           /\ UNCHANGED vars  
61  
62 Next == proc1 \vee proc2  
       \vee Terminating  
63  
64 Spec == /\ Init /\ [] [Next]_vars  
65           /\ WF_vars(proc1)  
66           /\ WF_vars(proc2)  
67  
68  
69 Termination == <>(\A self \in ProcSet: pc[self] = "Done")
```

PlusCal

Трансляция

```
58 (* Allow infinite stuttering to prevent deadlock on termination.  
   ↵ *)  
59 Terminating == /\ \A self \in ProcSet: pc[self] = "Done"  
60           /\ UNCHANGED vars  
61  
62 Next == proc1 \vee proc2  
       \vee Terminating  
63  
64 Spec == /\ Init /\ [] [Next]_vars  
          /\ WF_vars(proc1)  
          /\ WF_vars(proc2)  
65  
66  
67  
68  
69 Termination == <>(\A self \in ProcSet: pc[self] = "Done")
```

PlusCal

Трансляция

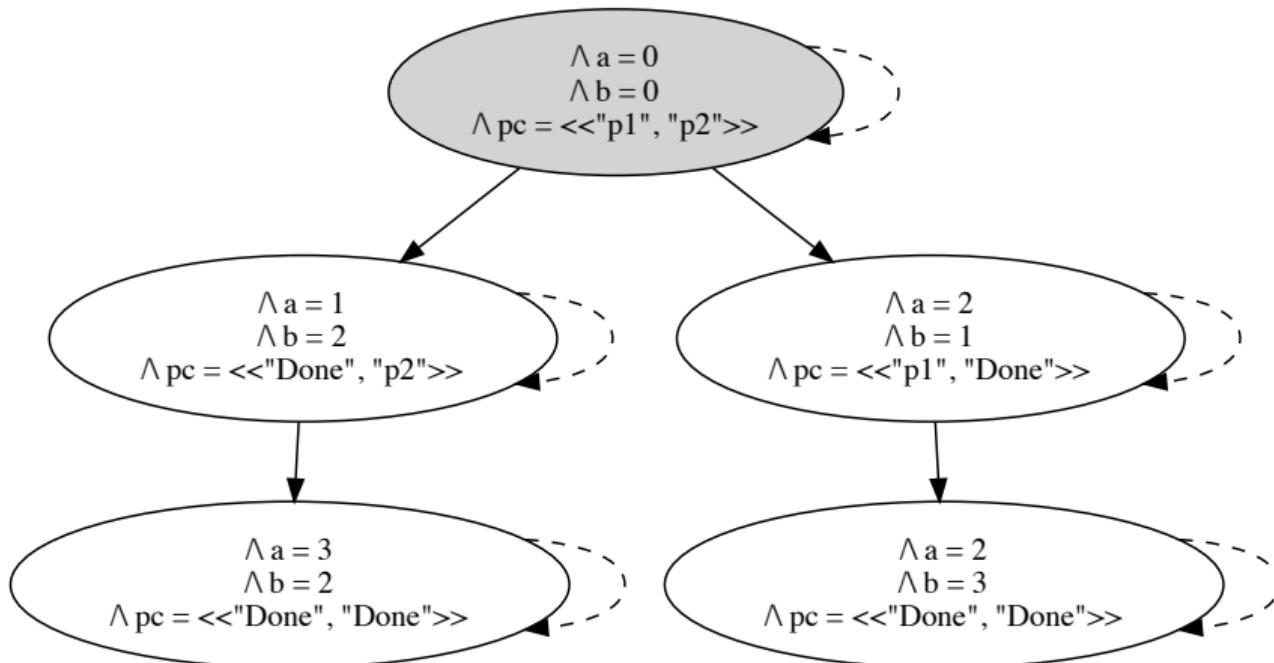
```
58 (* Allow infinite stuttering to prevent deadlock on termination.  
   ↵ *)  
59 Terminating == /\ \A self \in ProcSet: pc[self] = "Done"  
60           /\ UNCHANGED vars  
61  
62 Next == proc1 \vee proc2  
       \vee Terminating  
63  
64 Spec == /\ Init /\ [] [Next]_vars  
65     /\ WF_vars(proc1)  
66     /\ WF_vars(proc2)  
67  
68  
69 Termination == <>(\A self \in ProcSet: pc[self] = "Done")
```

ДЕМО

01_PCDemo.tla

TLA⁺ Toolbox CLI

```
growler@macbook:$ tlc -dump dot,colorized 01_PCDemo.dot 01_PCDemo.tla
Computing initial states...
Finished computing initial states: 1 distinct state generated.
Checking 2 branches of temporal properties for the complete state space
with 10 total distinct states
Finished checking temporal properties
Model checking completed. No error has been found.
Estimates of the probability that TLC did not check all reachable states
because two distinct states had the same fingerprint:
calculated (optimistic): val = 5.4E-19
7 states generated, 5 distinct states found, 0 states left on queue.
growler@macbook:$ dot -Tsvg 01_PCDemo.dot -o 01_PCDemo.svg
growler@macbook:$
```



PlusCal

Атомарность

```
4 (*--algorithm A
5
6 variables
7   a = 0,
8   b = 0;
9
10 define
11   Success == <>[ ](
12     (a + b) = 5
13   )
14 end define;

16   fair process proc1 = 1
17 begin
18   p1_1: a := b + 1;
19   p1_2: b := a + 1;
20 end process;

21
22   fair process proc2 = 2
23 begin
24   p2_1: b := a + 1;
25   p2_2: a := b + 1;
26 end process;

27
28 end algorithm; *)
```

Error: Temporal properties were violated.

Error: The following behavior constitutes a counter-example:

State 1: <Initial predicate>

```
/\ a = 0  
\ b = 0  
\ pc = <<"p1_1", "p2_1">>
```

State 2: <p1_1 line 48, col 9 to line 51, col 17 of module 02_PCDemo>

```
/\ a = 1  
\ b = 0  
\ pc = <<"p1_2", "p2_1">>
```

State 3: <p2_1 line 60, col 9 to line 63, col 17 of module 02_PCDemo>

```
/\ a = 1  
\ b = 2  
\ pc = <<"p1_2", "p2_2">>
```

State 4: <p2_2 line 65, col 9 to line 68, col 17 of module 02_PCDemo>

```
/\ a = 3  
\ b = 2  
\ pc = <<"p1_2", "Done">>
```

State 5: <p1_2 line 53, col 9 to line 56, col 17 of module 02_PCDemo>

```
/\ a = 3  
\ b = 4  
\ pc = <<"Done", "Done">>
```

Error: Temporal properties were violated.

Error: The following behavior constitutes a counter-example:

State 1: <Initial predicate>

```
/\ a = 0  
\ b = 0  
\ pc = <<"p1_1", "p2_1">>
```

State 2: <p1_1 line 48, col 9 to line 51, col 17 of module 02_PCDemo>

```
/\ a = 1  
\ b = 0  
\ pc = <<"p1_2", "p2_1">>
```

State 3: <p2_1 line 60, col 9 to line 63, col 17 of module 02_PCDemo>

```
/\ a = 1  
\ b = 2  
\ pc = <<"p1_2", "p2_2">>
```

State 4: <p2_2 line 65, col 9 to line 68, col 17 of module 02_PCDemo>

```
/\ a = 3  
\ b = 2  
\ pc = <<"p1_2", "Done">>
```

State 5: <p1_2 line 53, col 9 to line 56, col 17 of module 02_PCDemo>

```
/\ a = 3  
\ b = 4  
\ pc = <<"Done", "Done">>
```

Error: Temporal properties were violated.

Error: The following behavior constitutes a counter-example:

State 1: <Initial predicate>

```
/\ a = 0  
\ b = 0  
\ pc = <<"p1_1", "p2_1">>
```

State 2: <p1_1 line 48, col 9 to line 51, col 17 of module 02_PCDemo>

```
/\ a = 1  
\ b = 0  
\ pc = <<"p1_2", "p2_1">>
```

State 3: <p2_1 line 60, col 9 to line 63, col 17 of module 02_PCDemo>

```
/\ a = 1  
\ b = 2  
\ pc = <<"p1_2", "p2_2">>
```

State 4: <p2_2 line 65, col 9 to line 68, col 17 of module 02_PCDemo>

```
/\ a = 3  
\ b = 2  
\ pc = <<"p1_2", "Done">>
```

State 5: <p1_2 line 53, col 9 to line 56, col 17 of module 02_PCDemo>

```
/\ a = 3  
\ b = 4  
\ pc = <<"Done", "Done">>
```

Error: Temporal properties were violated.

Error: The following behavior constitutes a counter-example:

State 1: <Initial predicate>

```
/\ a = 0  
\ b = 0  
\ pc = <<"p1_1", "p2_1">>
```

State 2: <p1_1 line 48, col 9 to line 51, col 17 of module 02_PCDemo>

```
/\ a = 1  
\ b = 0  
\ pc = <<"p1_2", "p2_1">>
```

State 3: <p2_1 line 60, col 9 to line 63, col 17 of module 02_PCDemo>

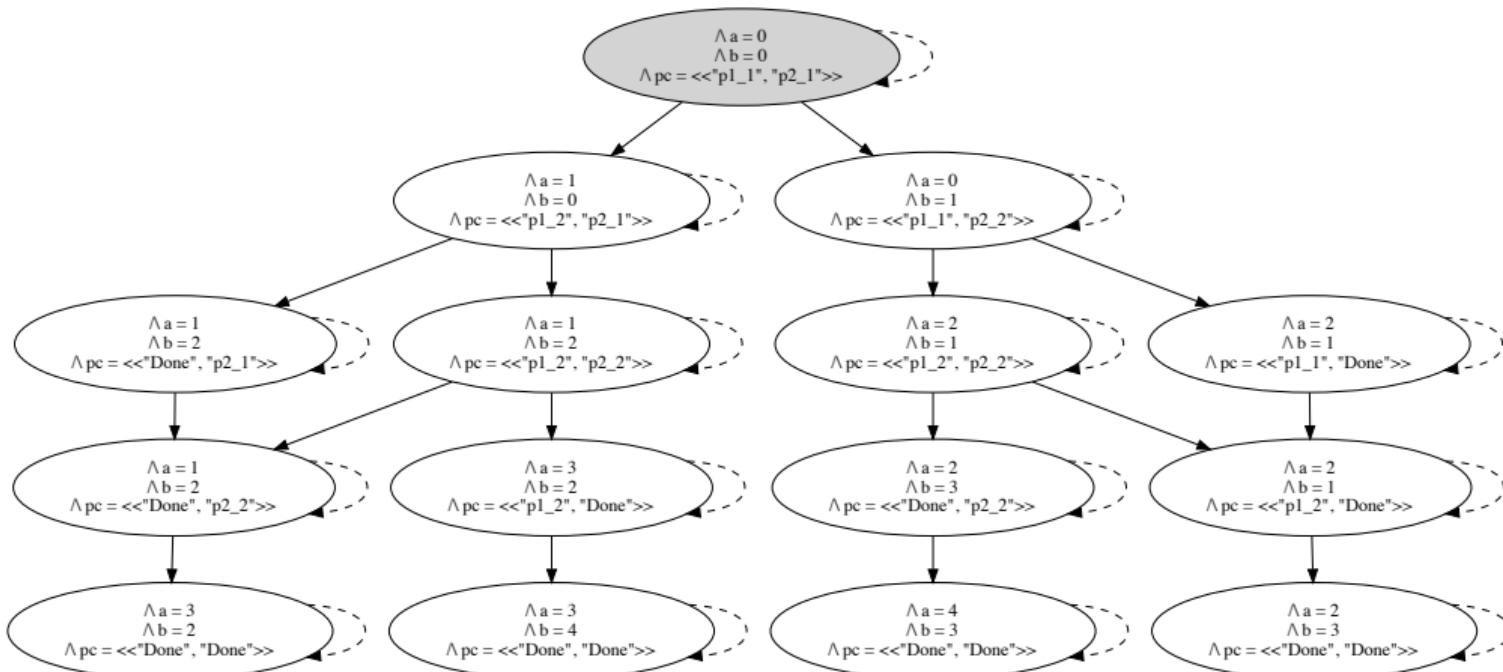
```
/\ a = 1  
\ b = 2  
\ pc = <<"p1_2", "p2_2">>
```

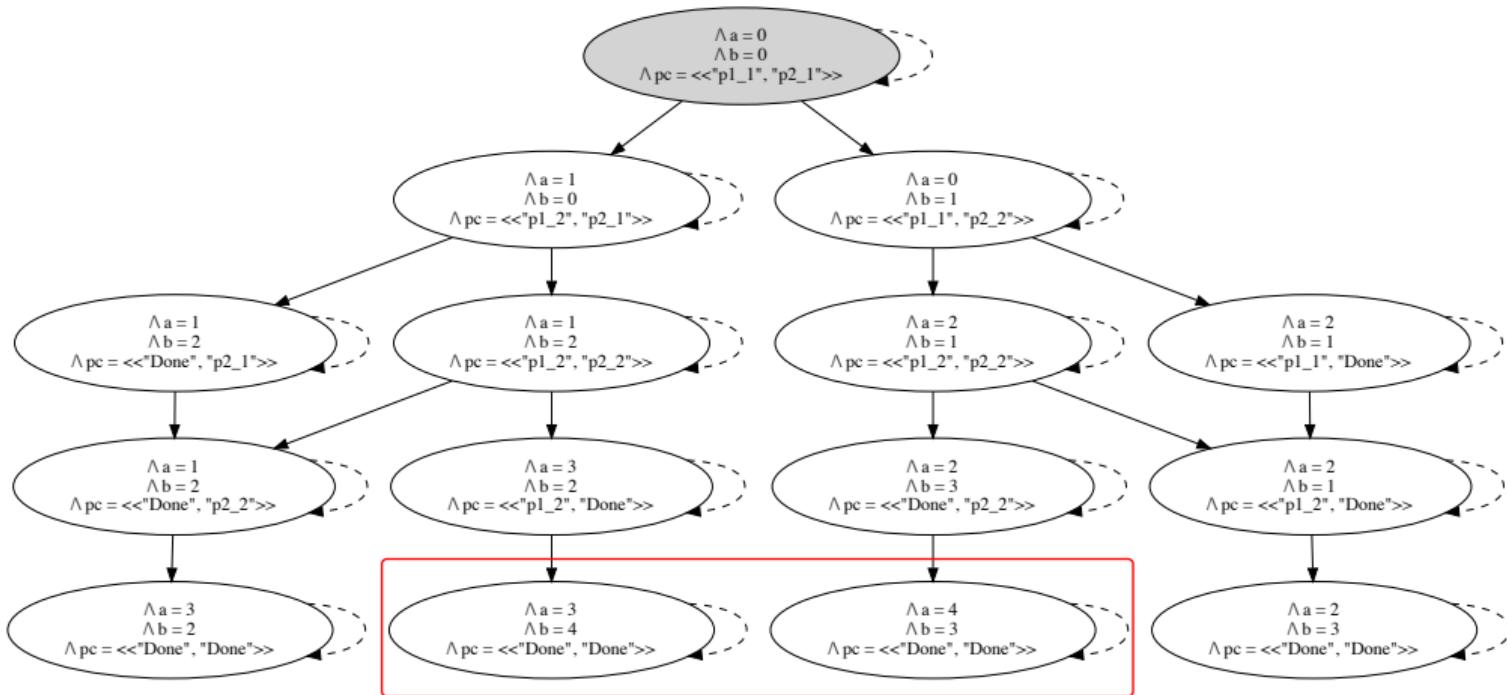
State 4: <p2_2 line 65, col 9 to line 68, col 17 of module 02_PCDemo>

```
/\ a = 3  
\ b = 2  
\ pc = <<"p1_2", "Done">>
```

State 5: <p1_2 line 53, col 9 to line 56, col 17 of module 02_PCDemo>

```
/\ a = 3  
\ b = 4  
\ pc = <<"Done", "Done">>
```





TLA+/PlusCal

Sets

```
1 Procs == {1, 5, 8}
2 Ops == {"inc", "dec"}
3 Vals == 1..MaxValue
4 Cmd == Ops \union {"nop"}
5 {1, 2, 3} \ {2} = {1, 3}
6 {1, 2} \intersect {2, 3} = {2}
7 var \in 1..10
8 ASSUME Var \notin Procs
9 1 \in Nat = TRUE
```

TLA+/PlusCal

Sets

```
1 Procs == {1, 5, 8}
2 Ops == {"inc", "dec"}
3 Vals == 1..MaxValue
4 Cmd == Ops \union {"nop"}
5 {1, 2, 3} \ {2} = {1, 3}
6 {1, 2} \intersect {2, 3} = {2}
7 var \in 1..10
8 ASSUME Var \notin Procs
9 1 \in Nat = TRUE
```

TLA+/PlusCal

Sets

```
1 Procs == {1, 5, 8}
2 Ops == {"inc", "dec"}
3 Vals == 1..MaxValue
4 Cmd == Ops \union {"nop"}
5 {1, 2, 3} \ {2} = {1, 3}
6 {1, 2} \intersect {2, 3} = {2}
7 var \in 1..10
8 ASSUME Var \notin Procs
9 1 \in Nat = TRUE
```

TLA+/PlusCal

Sets

```
1 Procs == {1, 5, 8}
2 Ops == {"inc", "dec"}
3 Vals == 1..MaxValue
4 Cmd == Ops \union {"nop"}  
{1, 2, 3} \ {2} = {1, 3}
6 {1, 2} \intersect {2, 3} = {2}
7 var \in 1..10
8 ASSUME Var \notin Procs
9 1 \in Nat = TRUE
```

TLA+/PlusCal

Sets

```
1 Procs == {1, 5, 8}
2 Ops == {"inc", "dec"}
3 Vals == 1..MaxValue
4 Cmd == Ops \union {"nop"}
5 {1, 2, 3} \setminus {2} = {1, 3}
6 {1, 2} \intersect {2, 3} = {2}
7 var \in 1..10
8 ASSUME Var \notin Procs
9 1 \in Nat = TRUE
```

TLA+/PlusCal

Sets

```
1 Procs == {1, 5, 8}
2 Ops == {"inc", "dec"}
3 Vals == 1..MaxValue
4 Cmd == Ops \union {"nop"}
5 {1, 2, 3} \setminus {2} = {1, 3}
6 {1, 2} \intersect {2, 3} = {2}
7 var \in 1..10
8 ASSUME Var \notin Procs
9 1 \in Nat = TRUE
```

TLA+/PlusCal

Sets

```
1 Procs == {1, 5, 8}
2 Ops == {"inc", "dec"}
3 Vals == 1..MaxValue
4 Cmd == Ops \union {"nop"}
5 {1, 2, 3} \ {2} = {1, 3}
6 {1, 2} \intersect {2, 3} = {2}
7 var \in 1..10
8 ASSUME Var \notin Procs
9 1 \in Nat = TRUE
```

TLA+/PlusCal

Sets

```
1 Procs == {1, 5, 8}
2 Ops == {"inc", "dec"}
3 Vals == 1..MaxValue
4 Cmd == Ops \union {"nop"}
5 {1, 2, 3} \ {2} = {1, 3}
6 {1, 2} \intersect {2, 3} = {2}
7 var \in 1..10
8 ASSUME Var \notin Procs
9 1 \in Nat = TRUE
```

TLA+/PlusCal Functions

```
1 Squares == [n \in Nat |-> n * n]
2 Squares[2] = 4
3 DOMAIN Squares = Nat
4 F == [n \in 1..3 |-> n * n]
5 DOMAIN F = 1..3
6 Q == [F EXCEPT ![2] = 0]
7 {Q[i]: i \in DOMAIN Q} = {1, 0, 9}
```

TLA+/PlusCal Functions

```
1 Squares == [n \in Nat |-> n * n]
2 Squares[2] = 4
3 DOMAIN Squares = Nat
4 F == [n \in 1..3 |-> n * n]
5 DOMAIN F = 1..3
6 Q == [F EXCEPT ![2] = 0]
7 {Q[i]: i \in DOMAIN Q} = {1, 0, 9}
```

TLA+/PlusCal Functions

```
1 Squares == [n \in Nat |-> n * n]
2 Squares[2] = 4
3 DOMAIN Squares = Nat
4 F == [n \in 1..3 |-> n * n]
5 DOMAIN F = 1..3
6 Q == [F EXCEPT ![2] = 0]
7 {Q[i]: i \in DOMAIN Q} = {1, 0, 9}
```

TLA+/PlusCal Functions

```
1 Squares == [n \in Nat |-> n * n]
2 Squares[2] = 4
3 DOMAIN Squares = Nat
4 F == [n \in 1..3 |-> n * n]
5 DOMAIN F = 1..3
6 Q == [F EXCEPT ![2] = 0]
7 {Q[i]: i \in DOMAIN Q} = {1, 0, 9}
```

TLA+/PlusCal Functions

```
1 Squares == [n \in Nat |-> n * n]
2 Squares[2] = 4
3 DOMAIN Squares = Nat
4 F == [n \in 1..3 |-> n * n]
5 DOMAIN F = 1..3
6 Q == [F EXCEPT ![2] = 0]
7 {Q[i]: i \in DOMAIN Q} = {1, 0, 9}
```

TLA+/PlusCal

Tuples

```
1 a == <<1, 5, 8>>
2 DOMAIN a = 1..3
3 a[2] = 5
4 Len(a) = 3
5 Head(a) = 1
6 Tail(a) = <<5, 8>>
7 SubSeq(a, 1, 2) = <<1, 5>>
8 SelectSeq(a, LAMBDA i: i > 5) = <<8>>
9 <<1, 2>> \o <<4, 8>> = <<1, 2, 4, 8>>
10 a := Append(a, 5)
11 a[i] := i
```

TLA+/PlusCal

Tuples

```
1 a == <<1, 5, 8>>
2 DOMAIN a = 1..3
3 a[2] = 5
4 Len(a) = 3
5 Head(a) = 1
6 Tail(a) = <<5, 8>>
7 SubSeq(a, 1, 2) = <<1, 5>>
8 SelectSeq(a, LAMBDA i: i > 5) = <<8>>
9 <<1, 2>> \o <<4, 8>> = <<1, 2, 4, 8>>
10 a := Append(a, 5)
11 a[i] := i
```

TLA+/PlusCal

Tuples

```
1 a == <<1, 5, 8>>
2 DOMAIN a = 1..3
3 a[2] = 5
4 Len(a) = 3
5 Head(a) = 1
6 Tail(a) = <<5, 8>>
7 SubSeq(a, 1, 2) = <<1, 5>>
8 SelectSeq(a, LAMBDA i: i > 5) = <<8>>
9 <<1, 2>> \o <<4, 8>> = <<1, 2, 4, 8>>
10 a := Append(a, 5)
11 a[i] := i
```

TLA+/PlusCal

Tuples

```
1 a == <<1, 5, 8>>
2 DOMAIN a = 1..3
3 a[2] = 5
4 Len(a) = 3
5 Head(a) = 1
6 Tail(a) = <<5, 8>>
7 SubSeq(a, 1, 2) = <<1, 5>>
8 SelectSeq(a, LAMBDA i: i > 5) = <<8>>
9 <<1, 2>> \o <<4, 8>> = <<1, 2, 4, 8>>
10 a := Append(a, 5)
11 a[i] := i
```

TLA+/PlusCal

Tuples

```
1 a == <<1, 5, 8>>
2 DOMAIN a = 1..3
3 a[2] = 5
4 Len(a) = 3
5 Head(a) = 1
6 Tail(a) = <<5, 8>>
7 SubSeq(a, 1, 2) = <<1, 5>>
8 SelectSeq(a, LAMBDA i: i > 5) = <<8>>
9 <<1, 2>> \o <<4, 8>> = <<1, 2, 4, 8>>
10 a := Append(a, 5)
11 a[i] := i
```

TLA+/PlusCal

Tuples

```
1 a == <<1, 5, 8>>
2 DOMAIN a = 1..3
3 a[2] = 5
4 Len(a) = 3
5 Head(a) = 1
6 Tail(a) = <<5, 8>>
7 SubSeq(a, 1, 2) = <<1, 5>>
8 SelectSeq(a, LAMBDA i: i > 5) = <<8>>
9 <<1, 2>> \o <<4, 8>> = <<1, 2, 4, 8>>
10 a := Append(a, 5)
11 a[i] := i
```

TLA+/PlusCal

Tuples

```
1 a == <<1, 5, 8>>
2 DOMAIN a = 1..3
3 a[2] = 5
4 Len(a) = 3
5 Head(a) = 1
6 Tail(a) = <<5, 8>>
7 SubSeq(a, 1, 2) = <<1, 5>>
8 SelectSeq(a, LAMBDA i: i > 5) = <<8>>
9 <<1, 2>> \o <<4, 8>> = <<1, 2, 4, 8>>
10 a := Append(a, 5)
11 a[i] := i
```

TLA+/PlusCal

Tuples

```
1 a == <<1, 5, 8>>
2 DOMAIN a = 1..3
3 a[2] = 5
4 Len(a) = 3
5 Head(a) = 1
6 Tail(a) = <<5, 8>>
7 SubSeq(a, 1, 2) = <<1, 5>>
8 SelectSeq(a, LAMBDA i: i > 5) = <<8>>
9 <<1, 2>> \o <<4, 8>> = <<1, 2, 4, 8>>
10 a := Append(a, 5)
11 a[i] := i
```

TLA+/PlusCal

Tuples

```
1 a == <<1, 5, 8>>
2 DOMAIN a = 1..3
3 a[2] = 5
4 Len(a) = 3
5 Head(a) = 1
6 Tail(a) = <<5, 8>>
7 SubSeq(a, 1, 2) = <<1, 5>>
8 SelectSeq(a, LAMBDA i: i > 5) = <<8>>
9 <<1, 2>> \o <<4, 8>> = <<1, 2, 4, 8>>
10 a := Append(a, 5)
11 a[i] := i
```

TLA+/PlusCal

Tuples

```
1 a == <<1, 5, 8>>
2 DOMAIN a = 1..3
3 a[2] = 5
4 Len(a) = 3
5 Head(a) = 1
6 Tail(a) = <<5, 8>>
7 SubSeq(a, 1, 2) = <<1, 5>>
8 SelectSeq(a, LAMBDA i: i > 5) = <<8>>
9 <<1, 2>> \o <<4, 8>> = <<1, 2, 4, 8>>
10 a := Append(a, 5)
11 a[i] := i
```

TLA+/PlusCal

Records

```
1 s == [rdy|->0, ack|->0, val|->0]
2 DOMAIN s = {"rdy", "ack", "val"}
3 s.rdy = 0
4 s["ack"] = 0
5 "a" :> 1 = [a|->1]
6 [a->1] @@ [b|->2] = [a|->1, b|->2]
7 [s EXCEPT !.rdy = @ + 1] = [rdy->1, ack|->0, val|->0]
8 s.ack := 1
9 [i \in DOMAIN s \ { "ack" } |-> s[i]] = [rdy|->0, val|->0]
```

TLA+/PlusCal

Records

```
1 s == [rdy|->0, ack|->0, val|->0]
2 DOMAIN s = {"rdy", "ack", "val"}
3 s.rdy = 0
4 s["ack"] = 0
5 "a" :> 1 = [a|->1]
6 [a->1] @@ [b|->2] = [a|->1, b|->2]
7 [s EXCEPT !.rdy = @ + 1] = [rdy->1, ack|->0, val|->0]
8 s.ack := 1
9 [i \in DOMAIN s \ { "ack" } |-> s[i]] = [rdy|->0, val|->0]
```

TLA+/PlusCal Records

```
1 s == [rdy|->0, ack|->0, val|->0]
2 DOMAIN s = {"rdy", "ack", "val"}
3 s.rdy = 0
4 s["ack"] = 0
5 "a" :> 1 = [a|->1]
6 [a->1] @@ [b|->2] = [a|->1, b|->2]
7 [s EXCEPT !.rdy = @ + 1] = [rdy->1, ack|->0, val|->0]
8 s.ack := 1
9 [i \in DOMAIN s \ { "ack" } |-> s[i]] = [rdy|->0, val|->0]
```

TLA+/PlusCal Records

```
1 s == [rdy|->0, ack|->0, val|->0]
2 DOMAIN s = {"rdy", "ack", "val"}
3 s.rdy = 0
4 s["ack"] = 0
5 "a" :> 1 = [a|->1]
6 [a->1] @@ [b|->2] = [a|->1, b|->2]
7 [s EXCEPT !.rdy = @ + 1] = [rdy->1, ack|->0, val|->0]
8 s.ack := 1
9 [i \in DOMAIN s \ { "ack" } |-> s[i]] = [rdy|->0, val|->0]
```

TLA+/PlusCal

Records

```
1 s == [rdy|->0, ack|->0, val|->0]
2 DOMAIN s = {"rdy", "ack", "val"}
3 s.rdy = 0
4 s["ack"] = 0
5 "a" :> 1 = [a|->1]
6 [a->1] @ [b|->2] = [a|->1, b|->2]
7 [s EXCEPT !.rdy = @ + 1] = [rdy->1, ack|->0, val|->0]
8 s.ack := 1
9 [i \in DOMAIN s \ {"ack"} |-> s[i]] = [rdy|->0, val|->0]
```

TLA+/PlusCal Records

```
1 s == [rdy|->0, ack|->0, val|->0]
2 DOMAIN s = {"rdy", "ack", "val"}
3 s.rdy = 0
4 s["ack"] = 0
5 "a" :> 1 = [a|->1]
6 [a->1] @@ [b|->2] = [a|->1, b|->2]
7 [s EXCEPT !.rdy = @ + 1] = [rdy->1, ack|->0, val|->0]
8 s.ack := 1
9 [i \in DOMAIN s \ { "ack" } |-> s[i]] = [rdy|->0, val|->0]
```

TLA+/PlusCal Records

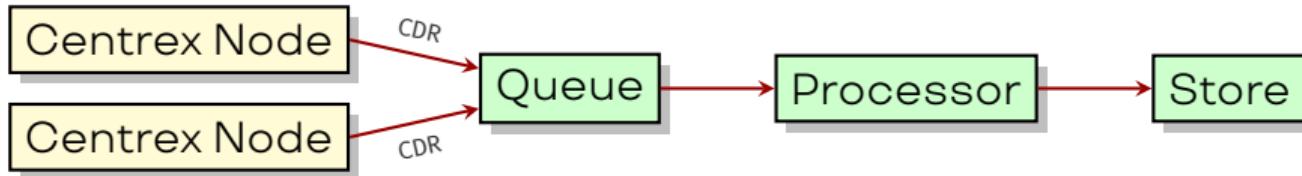
```
1 s == [rdy|->0, ack|->0, val|->0]
2 DOMAIN s = {"rdy", "ack", "val"}
3 s.rdy = 0
4 s["ack"] = 0
5 "a" :> 1 = [a|->1]
6 [a->1] @@ [b|->2] = [a|->1, b|->2]
7 [s EXCEPT !.rdy = @ + 1] = [rdy->1, ack|->0, val|->0]
8 s.ack := 1
9 [i \in DOMAIN s \ { "ack" } |-> s[i]] = [rdy|->0, val|->0]
```

TLA+/PlusCal

Records

```
1 s == [rdy|->0, ack|->0, val|->0]
2 DOMAIN s = {"rdy", "ack", "val"}
3 s.rdy = 0
4 s["ack"] = 0
5 "a" :> 1 = [a|->1]
6 [a->1] @@ [b|->2] = [a|->1, b|->2]
7 [s EXCEPT !.rdy = @ + 1] = [rdy->1, ack|->0, val|->0]
8 s.ack := 1
9 [i \in DOMAIN s \ { "ack" } |-> s[i]] = [rdy|->0, val|->0]
```

Задача



Модель: требования

CallHistory.tla v0

```
1 ----- MODULE CallHistory -----
2 EXTENDS Sequences, Integers, TLC
3
4 CONSTANTS Calls
5
6 (*--algorithm CallHistory
7
8 variables
9   queue = <<>>,
10  store = <<>>;
11
12 define
13
14   AllCallsProcessed == <>[ ](
15     \A i \in Calls:
16       Len(SelectSeq(store, LAMBDA
17         \B j: j = i)) = 1
18   )
19
20   Success == AllCallsProcessed
21
22 end define;
```

```
23   fair process Call \in Calls
24   begin CallSetup:
25     skip;
26   end process;
27
28   fair process Processor = "Processor"
29   begin Process:
30     while TRUE do
31       ProcessCDR:
32         skip;
33       end while;
34     end process;
35
36   end algorithm; *)
```

CallHistory.tla v0

```
1 ----- MODULE CallHistory -----
2 EXTENDS Sequences, Integers, TLC
3
4 CONSTANTS Calls
5
6 (*--algorithm CallHistory
7
8 variables
9   queue = <<>>;  
10  store = <<>>;  
11
12 define
13
14   AllCallsProcessed == <>[ ](  
15     \A i \in Calls:  
16       Len(SelectSeq(store, LAMBDA  
17         \j: j = i)) = 1
18   )
19
20   Success == AllCallsProcessed
21
22 end define;
```

```
23   fair process Call \in Calls
24   begin CallSetup:
25     skip;
26   end process;
27
28   fair process Processor = "Processor"
29   begin Process:
30     while TRUE do
31       ProcessCDR:
32         skip;
33     end while;
34   end process;
35
36 end algorithm; *)
```

CallHistory.tla v0

```
1 ----- MODULE CallHistory -----
2 EXTENDS Sequences, Integers, TLC
3
4 CONSTANTS Calls
5
6 (*--algorithm CallHistory
7
8 variables
9   queue = <<>>,
10  store = <<>>;
11
12 define
13
14   AllCallsProcessed == <>[ ](
15     \A i \in Calls:
16       Len(SelectSeq(store, LAMBDA
17         \B j: j = i)) = 1
18
19   Success == AllCallsProcessed
20
21 end define;
22
```

```
23   fair process Call \in Calls
24   begin CallSetup:
25     skip;
26   end process;
27
28   fair process Processor = "Processor"
29   begin Process:
30     while TRUE do
31       ProcessCDR:
32         skip;
33       end while;
34     end process;
35
36   end algorithm; *)
```

CallHistory.tla v0

```
1 ----- MODULE CallHistory -----
2 EXTENDS Sequences, Integers, TLC
3
4 CONSTANTS Calls
5
6 (*--algorithm CallHistory
7
8 variables
9   queue = <<>>,
10  store = <<>>;
11
12 define
13
14   AllCallsProcessed == <>[ ](
15     \A i \in Calls:
16       Len(SelectSeq(store, LAMBDA
17         \B j: j = i)) = 1
18   )
19
20   Success == AllCallsProcessed
21
22 end define;
```

```
23   fair process Call \in Calls
24   begin CallSetup:
25     skip;
26   end process;
27
28   fair process Processor = "Processor"
29   begin Process:
30     while TRUE do
31       ProcessCDR:
32         skip;
33       end while;
34     end process;
35
36   end algorithm; *)
```

CallHistory.tla v0

```
1 ----- MODULE CallHistory -----
2 EXTENDS Sequences, Integers, TLC
3
4 CONSTANTS Calls
5
6 (*--algorithm CallHistory
7
8 variables
9   queue = <<>>,
10  store = <<>>;
11
12 define
13
14   AllCallsProcessed == <>[ ](
15     \A i \in Calls:
16       Len(SelectSeq(store, LAMBDA
17         \B j: j = i)) = 1
18   )
19
20   Success == AllCallsProcessed
21
22 end define;
```

```
23   fair process Call \in Calls
24   begin CallSetup:
25     skip;
26   end process;
27
28   fair process Processor = "Processor"
29   begin Process:
30     while TRUE do
31       ProcessCDR:
32         skip;
33       end while;
34     end process;
35
36   end algorithm; *)
```

CallHistory.tla v0

```
1 ----- MODULE CallHistory -----
2 EXTENDS Sequences, Integers, TLC
3
4 CONSTANTS Calls
5
6 (*--algorithm CallHistory
7
8 variables
9   queue = <>>;,
10  store = <>>;
11
12 define
13
14   AllCallsProcessed == <>[ ](
15     \A i \in Calls:
16       Len(SelectSeq(store, LAMBDA
17         \B j: j = i)) = 1
18   )
19
20   Success == AllCallsProcessed
21
22 end define;
```

```
23   fair process Call \in Calls
24   begin CallSetup:
25     skip;
26   end process;
27
28   fair process Processor = "Processor"
29   begin Process:
30     while TRUE do
31       ProcessCDR:
32         skip;
33       end while;
34   end process;
35
36 end algorithm; *)
```

CallHistory.tla v0

```
1 ----- MODULE CallHistory -----
2 EXTENDS Sequences, Integers, TLC
3
4 CONSTANTS Calls
5
6 (*--algorithm CallHistory
7
8 variables
9   queue = <<>>,
10  store = <<>>;
11
12 define
13
14   AllCallsProcessed == <>[ ](
15     \A i \in Calls:
16       Len(SelectSeq(store, LAMBDA
17         \B j: j = i)) = 1
18   )
19
20   Success == AllCallsProcessed
21
22 end define;
```

```
23   fair process Call \in Calls
24   begin CallSetup:
25     skip;
26   end process;
27
28   fair process Processor = "Processor"
29   begin Process:
30     while TRUE do
31       ProcessCDR:
32         skip;
33       end while;
34   end process;
35
36 end algorithm; *)
```

CallHistory.cfg v0

```
1 SPECIFICATION Spec
2 /* Add statements after this line.
3
4 CONSTANTS
5     Calls = {"call-1", "call-2", "call-3"}
6
7 PROPERTIES
8     Success
```

CallHistory.cfg v0

```
1 SPECIFICATION Spec
2 /* Add statements after this line.
3
4 CONSTANTS
5     Calls = {"call-1", "call-2", "call-3"}
6
7 PROPERTIES
8     Success
```

CallHistory.tla v0 Run: Failed

```
19 Progress(5) at 2021-09-15 20:52:53: 41 states generated, 16 distinct states found, 0
  ↵ states left on queue.
20 Checking temporal properties for the complete state space with 16 total distinct states
  ↵ at (2021-09-15 20:52:53)
21 Error: Temporal properties were violated.
22
23 Error: The following behavior constitutes a counter-example:
64
65 State 6: <ProcessCDR line 70, col 15 to line 73, col 45 of module CallHistory>
66 /\ store = <>>
67 /\ pc = ( "call-1" :> "Done" ∧
68   "call-2" :> "Done" ∧
69   "call-3" :> "Done" ∧
70   "Processor" :> "Process" )
71 /\ queue = <>>
72
73 Back to state 5: <Process line 66, col 12 to line 68, col 42 of module CallHistory>
74
75 Finished checking temporal properties in 00s at 2021-09-15 20:52:53
76 41 states generated, 16 distinct states found, 0 states left on queue.
```

CallHistory.tla v0 Run: Failed

```
19 Progress(5) at 2021-09-15 20:52:53: 41 states generated, 16 distinct states found, 0
  ↵ states left on queue.
20 Checking temporal properties for the complete state space with 16 total distinct states
  ↵ at (2021-09-15 20:52:53)
21 Error: Temporal properties were violated.
22
23 Error: The following behavior constitutes a counter-example:
64
65 State 6: <ProcessCDR line 70, col 15 to line 73, col 45 of module CallHistory>
66 /\ store = <>>
67 /\ pc = ( "call-1" :> "Done" ∧
68   "call-2" :> "Done" ∧
69   "call-3" :> "Done" ∧
70   "Processor" :> "Process" )
71 /\ queue = <>>
72
73 Back to state 5: <Process line 66, col 12 to line 68, col 42 of module CallHistory>
74
75 Finished checking temporal properties in 00s at 2021-09-15 20:52:53
76 41 states generated, 16 distinct states found, 0 states left on queue.
```

CallHistory.tla v0 Run: Failed

```
19 Progress(5) at 2021-09-15 20:52:53: 41 states generated, 16 distinct states found, 0
  ↵ states left on queue.
20 Checking temporal properties for the complete state space with 16 total distinct states
  ↵ at (2021-09-15 20:52:53)
21 Error: Temporal properties were violated.
22
23 Error: The following behavior constitutes a counter-example:
64
65 State 6: <ProcessCDR line 70, col 15 to line 73, col 45 of module CallHistory>
66 /\ store = <>>
67 /\ pc = ( "call-1" :> "Done" ;&
68   "call-2" :> "Done" ;&
69   "call-3" :> "Done" ;&
70   "Processor" :> "Process" )
71 /\ queue = <>>>
72
73 Back to state 5: <Process line 66, col 12 to line 68, col 42 of module CallHistory>
74
75 Finished checking temporal properties in 00s at 2021-09-15 20:52:53
76 41 states generated, 16 distinct states found, 0 states left on queue.
```

CallHistory.tla v0 Run: Failed

```
19 Progress(5) at 2021-09-15 20:52:53: 41 states generated, 16 distinct states found, 0
  ↵ states left on queue.
20 Checking temporal properties for the complete state space with 16 total distinct states
  ↵ at (2021-09-15 20:52:53)
21 Error: Temporal properties were violated.
22
23 Error: The following behavior constitutes a counter-example:
64
65 State 6: <ProcessCDR line 70, col 15 to line 73, col 45 of module CallHistory>
66 /\ store = <>>
67 /\ pc = ( "call-1" :> "Done" ∧
68   "call-2" :> "Done" ∧
69   "call-3" :> "Done" ∧
70   "Processor" :> "Process" )
71 /\ queue = <>>
72
73 Back to state 5: <Process line 66, col 12 to line 68, col 42 of module CallHistory>
74
75 Finished checking temporal properties in 00s at 2021-09-15 20:52:53
76 41 states generated, 16 distinct states found, 0 states left on queue.
```

CallHistory.tla v0 Run: Failed

```
19 Progress(5) at 2021-09-15 20:52:53: 41 states generated, 16 distinct states found, 0
  ↵ states left on queue.
20 Checking temporal properties for the complete state space with 16 total distinct states
  ↵ at (2021-09-15 20:52:53)
21 Error: Temporal properties were violated.
22
23 Error: The following behavior constitutes a counter-example:
64
65 State 6: <ProcessCDR line 70, col 15 to line 73, col 45 of module CallHistory>
66 /\ store = <>>
67 /\ pc = ( "call-1" :> "Done" ;&
68   "call-2" :> "Done" ;&
69   "call-3" :> "Done" ;&
70   "Processor" :> "Process" )
71 /\ queue = <>>
72
73 Back to state 5: <Process line 66, col 12 to line 68, col 42 of module CallHistory>
74
75 Finished checking temporal properties in 00s at 2021-09-15 20:52:53
76 41 states generated, 16 distinct states found, 0 states left on queue.
```

Модель:
первая реализация

CallHistory.tla v0->v1

```
15  (*--algorithm CallHistory
16
32  fair process Call \in Calls
33  begin CallSetup:
34  -    skip;
35  +    queue := Append(queue, [id |-> self, action |-> "setup"]);
36  +CallComplete:
37  +    queue := Append(queue, [id |-> self, action |-> "complete"]);
38  end process;
39
```

CallHistory.tla v0->v1

```
15  (*--algorithm CallHistory
16
32  fair process Call \in Calls
33  begin CallSetup:
34  -    skip;
35  +    queue := Append(queue, [id |-> self, action |-> "setup"]);
36  +CallComplete:
37  +    queue := Append(queue, [id |-> self, action |-> "complete"]);
38  end process;
39
```

CallHistory.tla v0->v1

```
10  CONSTANTS Calls
11
12  +Add(dict, key, val) == dict @@ (key :> val)
13  +Remove(dict, key) == [i \in DOMAIN dict \ {key} |-> dict[i]]
14  +
15  (*--algorithm CallHistory
16
17  variables
18      queue = <>>,
19  +    committed = 0,
20      store = <>>;
21  define
22
23      AllCallsProcessed == <>[|(
24          \A i \in Calls:
25              Len(SelectSeq(store, LAMBDA j: j = i)) = 1
26      )
27
28      Success == AllCallsProcessed
29
30  end define;
```

CallHistory.tla v0->v1

```
10  CONSTANTS Calls
11
12 +Add(dict, key, val) == dict @> (key :> val)
13 +Remove(dict, key) == [i \in DOMAIN dict \ {key} |-> dict[i]]
14 +
15 (*--algorithm CallHistory
16
17 variables
18     queue = <>>,
19 +    committed = 0,
20     store = <>>;
21 define
22
23     AllCallsProcessed == <>[|(
24         \A i \in Calls:
25             Len(SelectSeq(store, LAMBDA j: j = i)) = 1
26     )
27
28     Success == AllCallsProcessed
29
30 end define;
```

CallHistory.tla v0->v1

```
10  CONSTANTS Calls
11
12  +Add(dict, key, val) == dict @@ (key :> val)
13  +Remove(dict, key) == [i \in DOMAIN dict \ {key} |-> dict[i]]
14 +
15 (*--algorithm CallHistory
16
17 variables
18   queue = <>>,
19 +  committed = 0,
20   store = <>>;
21 define
22
23   AllCallsProcessed == <>[|(
24     \A i \in Calls:
25       Len(SelectSeq(store, LAMBDA j: j = i)) = 1
26   )
27
28   Success == AllCallsProcessed
29
30 end define;
```

CallHistory.tla v0->v1

```
40  fair process Processor = "Processor"
41  +variables
42  +    wip = <>>,
43  +    offset = committed + 1;
44  begin Process:
45      while TRUE do
46          +      when Len(queue) >= offset;
47          ProcessCDR:
48              -      skip;
49              +      with cdr = queue[offset]
50              +      do
51                  +          if cdr.id \notin DOMAIN wip /\ cdr.action = "setup" then
52                      +              wip := Add(wip, cdr.id, [offset |-> offset]);
53                  +          elseif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then
54                      +              wip := Remove(wip, cdr.id);
55                      +              store := Append(store, cdr.id);
56                  +          end if;
57              +      end with;
58              +      committed := offset;
59              +      offset := offset + 1;
60      end while;
61  end process;
```

CallHistory.tla v0->v1

```
40  fair process Processor = "Processor"
41  +variables
42  +    wip = <>>,
43  +    offset = committed + 1;
44  begin Process:
45      while TRUE do
46          +      when Len(queue) >= offset;
47          ProcessCDR:
48          -      skip;
49          +      with cdr = queue[offset]
50          +      do
51              +                  if cdr.id \notin DOMAIN wip /\ cdr.action = "setup" then
52              +                      wip := Add(wip, cdr.id, [offset |-> offset]);
53              +                  elseif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then
54              +                      wip := Remove(wip, cdr.id);
55              +                      store := Append(store, cdr.id);
56              +                  end if;
57              +      end with;
58              +      committed := offset;
59              +      offset := offset + 1;
60      end while;
61  end process;
```

CallHistory.tla v0->v1

```
40  fair process Processor = "Processor"
41  +variables
42  +    wip = <>>,
43  +    offset = committed + 1;
44  begin Process:
45      while TRUE do
46          +      when Len(queue) >= offset;
47          ProcessCDR:
48              -      skip;
49              +      with cdr = queue[offset]
50              +      do
51                  +          if cdr.id \notin DOMAIN wip /\ cdr.action = "setup" then
52                      +              wip := Add(wip, cdr.id, [offset |-> offset]);
53                  +          elseif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then
54                      +              wip := Remove(wip, cdr.id);
55                      +              store := Append(store, cdr.id);
56                  +          end if;
57              +      end with;
58              +      committed := offset;
59              +      offset := offset + 1;
60      end while;
61  end process;
```

CallHistory.tla v0->v1

```
40  fair process Processor = "Processor"
41  +variables
42  +    wip = <>>,
43  +    offset = committed + 1;
44  begin Process:
45      while TRUE do
46          +      when Len(queue) >= offset;
47          ProcessCDR:
48              -      skip;
49              +      with cdr = queue[offset]
50                  do
51                      +              if cdr.id \notin DOMAIN wip /\ cdr.action = "setup" then
52                          +                  wip := Add(wip, cdr.id, [offset |-> offset]);
53                      +                      elseif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then
54                          +                              wip := Remove(wip, cdr.id);
55                          +                              store := Append(store, cdr.id);
56                      +                          end if;
57                  end with;
58                  +                  committed := offset;
59                  +                  offset := offset + 1;
60              end while;
61  end process;
```

CallHistory.tla v0->v1

```
40  fair process Processor = "Processor"
41  +variables
42  +    wip = <>>,
43  +    offset = committed + 1;
44  begin Process:
45      while TRUE do
46          +      when Len(queue) >= offset;
47          ProcessCDR:
48              -      skip;
49              +      with cdr = queue[offset]
50                  +      do
51                      +                      if cdr.id \notin DOMAIN wip /\ cdr.action = "setup" then
52                          +                          wip := Add(wip, cdr.id, [offset |-> offset]);
53                      +                      elseif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then
54                          +                          wip := Remove(wip, cdr.id);
55                          +                          store := Append(store, cdr.id);
56                      +                      end if;
57                  +                  end with;
58                  +                  committed := offset;
59                  +                  offset := offset + 1;
60              end while;
61  end process;
```

CallHistory.tla v0->v1

```
40  fair process Processor = "Processor"
41  +variables
42  +    wip = <>>,
43  +    offset = committed + 1;
44  begin Process:
45      while TRUE do
46          +      when Len(queue) >= offset;
47          ProcessCDR:
48              -      skip;
49              +      with cdr = queue[offset]
50                  +      do
51                      +                      if cdr.id \notin DOMAIN wip /\ cdr.action = "setup" then
52                          +                          wip := Add(wip, cdr.id, [offset |-> offset]);
53                      +                      elseif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then
54                          +                          wip := Remove(wip, cdr.id);
55                          +                          store := Append(store, cdr.id);
56                      +                      end if;
57                  +                  end with;
58                  +                  committed := offset;
59                  +                  offset := offset + 1;
60              end while;
61  end process;
```

CallHistory.tla v0->v1

```
40  fair process Processor = "Processor"
41  +variables
42  +    wip = <>>,
43  +    offset = committed + 1;
44  begin Process:
45      while TRUE do
46          +      when Len(queue) >= offset;
47          ProcessCDR:
48              -      skip;
49              +      with cdr = queue[offset]
50                  +      do
51                      +                      if cdr.id \notin DOMAIN wip /\ cdr.action = "setup" then
52                          +                          wip := Add(wip, cdr.id, [offset |-> offset]);
53                      +                      elseif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then
54                          +                              wip := Remove(wip, cdr.id);
55                          +                              store := Append(store, cdr.id);
56                      +                      end if;
57                  +                  end with;
58                  +                  committed := offset;
59                  +                  offset := offset + 1;
60              end while;
61  end process;
```

CallHistory.tla v0->v1

```
40  fair process Processor = "Processor"
41  +variables
42  +    wip = <>>,
43  +    offset = committed + 1;
44  begin Process:
45      while TRUE do
46          +      when Len(queue) >= offset;
47          ProcessCDR:
48              -      skip;
49              +      with cdr = queue[offset]
50              +      do
51                  +          if cdr.id \notin DOMAIN wip /\ cdr.action = "setup" then
52                      +              wip := Add(wip, cdr.id, [offset |-> offset]);
53                  +          elseif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then
54                      +                          wip := Remove(wip, cdr.id);
55                      +                          store := Append(store, cdr.id);
56                  +          end if;
57              +          end with;
58              +          committed := offset;
59              +          offset := offset + 1;
60      end while;
61  end process;
```

CallHistory.tla v0->v1

```
40  fair process Processor = "Processor"
41  +variables
42  +    wip = <>>,
43  +    offset = committed + 1;
44  begin Process:
45      while TRUE do
46          +      when Len(queue) >= offset;
47          ProcessCDR:
48              -      skip;
49              +      with cdr = queue[offset]
50              +      do
51                  +          if cdr.id \notin DOMAIN wip /\ cdr.action = "setup" then
52                      +              wip := Add(wip, cdr.id, [offset |-> offset]);
53                  +          elseif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then
54                      +              wip := Remove(wip, cdr.id);
55                  +                  store := Append(store, cdr.id);
56                  +              end if;
57              +          end with;
58              +          committed := offset;
59              +          offset := offset + 1;
60      end while;
61  end process;
```

CallHistory.tla v0->v1

```
40  fair process Processor = "Processor"
41  +variables
42  +    wip = <>>,
43  +    offset = committed + 1;
44  begin Process:
45      while TRUE do
46          +      when Len(queue) >= offset;
47          ProcessCDR:
48              -      skip;
49              +      with cdr = queue[offset]
50              +      do
51                  +          if cdr.id \notin DOMAIN wip /\ cdr.action = "setup" then
52                      +              wip := Add(wip, cdr.id, [offset |-> offset]);
53                  +          elseif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then
54                      +              wip := Remove(wip, cdr.id);
55                      +              store := Append(store, cdr.id);
56                  +          end if;
57              +      end with;
58              +      committed := offset;
59              +      offset := offset + 1;
60      end while;
61  end process;
```

CallHistory.tla v0->v1

```
40  fair process Processor = "Processor"
41  +variables
42  +    wip = <>>,
43  +    offset = committed + 1;
44  begin Process:
45      while TRUE do
46          +      when Len(queue) >= offset;
47          ProcessCDR:
48              -      skip;
49              +      with cdr = queue[offset]
50              +      do
51                  +          if cdr.id \notin DOMAIN wip /\ cdr.action = "setup" then
52                      +              wip := Add(wip, cdr.id, [offset |-> offset]);
53                  +          elseif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then
54                      +              wip := Remove(wip, cdr.id);
55                      +              store := Append(store, cdr.id);
56                  +          end if;
57              +      end with;
58              +      committed := offset;
59              +      offset := offset + 1;
60      end while;
61  end process;
```

CallHistory.tla v1 Run: Succeed

```
22 Model checking completed. No error has been found.  
23 Estimates of the probability that TLC did not check all reachable states  
24 because two distinct states had the same fingerprint:  
25 calculated (optimistic): val = 3.2E-13  
26 4927 states generated, 2869 distinct states found, 0 states left on queue.  
27 The depth of the complete state graph search is 19.  
28 The average outdegree of the complete state graph is 1 (minimum is 0, the maximum 4 and  
→ the 95th percentile is 3).
```

CallHistory.tla v1 Run: Succeed

```
22 Model checking completed. No error has been found.  
23 Estimates of the probability that TLC did not check all reachable states  
24 because two distinct states had the same fingerprint:  
25 calculated (optimistic): val = 3.2E-13  
26 4927 states generated, 2869 distinct states found, 0 states left on queue.  
27 The depth of the complete state graph search is 19.  
28 The average outdegree of the complete state graph is 1 (minimum is 0, the maximum 4 and  
→ the 95th percentile is 3).
```

Модель:
добавим graceful shutdown

CallHistory.tla v1->v2

```
15  (*--algorithm CallHistory
16
17  variables
18      queue = <>>,
19      committed = 0,
20 + shutdownProcessor = FALSE,
21      store = <>>;
22
23  fair process Call \in Calls
24  begin CallSetup:
25      queue := Append(queue, [id |-> self, action |-> "setup"]);
26  CallComplete:
27      queue := Append(queue, [id |-> self, action |-> "complete"]);
28  end process;
29
30 +fair process Shutdown = "Shutdown"
31 +begin SignalShutdown:
32 +    when \A c \in Calls: pc[c] = "Done";
33 +    shutdownProcessor := TRUE;
34 +end process;
```

CallHistory.tla v1->v2

```
15  (*--algorithm CallHistory
16
17  variables
18    queue = <>>,
19    committed = 0,
20 +  shutdownProcessor = FALSE,
21    store = <>>;
22
23  fair process Call \in Calls
24  begin CallSetup:
25    queue := Append(queue, [id |-> self, action |-> "setup"]);
26  CallComplete:
27    queue := Append(queue, [id |-> self, action |-> "complete"]);
28  end process;
29
30 +fair process Shutdown = "Shutdown"
31 +begin SignalShutdown:
32 +  when \A c \in Calls: pc[c] = "Done";
33 +  shutdownProcessor := TRUE;
34 +end process;
```

CallHistory.tla v1->v2

```
15  (*--algorithm CallHistory
16
17  variables
18    queue = <>>,
19    committed = 0,
20 +  shutdownProcessor = FALSE,
21    store = <>>;
22
23  fair process Call \in Calls
24  begin CallSetup:
25    queue := Append(queue, [id |-> self, action |-> "setup"]);
26  CallComplete:
27    queue := Append(queue, [id |-> self, action |-> "complete"]);
28  end process;
29
30 +fair process Shutdown = "Shutdown"
31 +begin SignalShutdown:
32 +  when \A c \in Calls: pc[c] = "Done";
33 +  shutdownProcessor := TRUE;
34 +end process;
```

CallHistory.tla v1->v2

```
15  (*--algorithm CallHistory
16
17  variables
18    queue = <>>,
19    committed = 0,
20 +  shutdownProcessor = FALSE,
21    store = <>>;
22
23  fair process Call \in Calls
24  begin CallSetup:
25    queue := Append(queue, [id |-> self, action |-> "setup"]);
26  CallComplete:
27    queue := Append(queue, [id |-> self, action |-> "complete"]);
28  end process;
29
30 +fair process Shutdown = "Shutdown"
31 +begin SignalShutdown:
32 +  when \A c \in Calls: pc[c] = "Done";
33 +  shutdownProcessor := TRUE;
34 +end process;
```

CallHistory.tla v1->v2

```
50 begin Process:  
51     while TRUE do  
52         either  
53             when Len(queue) >= offset;  
54             ProcessCDR:  
55                 with cdr = queue[offset]  
56                 do  
57                     if cdr.id \notin DOMAIN wip /\ cdr.action = "setup" then  
58                         wip := Add(wip, cdr.id, [offset |-> offset]);  
59                     elsif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then  
60                         wip := Remove(wip, cdr.id);  
61                         store := Append(store, cdr.id);  
62                     end if;  
63                 end with;  
64                 committed := offset;  
65                 offset := offset + 1;  
66         or  
67         when shutdownProcessor;  
68         goto Done;  
69     end either;  
70 end while;  
71 end process;
```

CallHistory.tla v1->v2

```
50 begin Process:  
51     while TRUE do  
52         either  
53             when Len(queue) >= offset;  
54             ProcessCDR:  
55                 with cdr = queue[offset]  
56                 do  
57                     if cdr.id \notin DOMAIN wip /\ cdr.action = "setup" then  
58                         wip := Add(wip, cdr.id, [offset |-> offset]);  
59                     elsif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then  
60                         wip := Remove(wip, cdr.id);  
61                         store := Append(store, cdr.id);  
62                     end if;  
63                 end with;  
64                 committed := offset;  
65                 offset := offset + 1;  
66         or  
67         when shutdownProcessor;  
68         goto Done;  
69     end either;  
70 end while;  
71 end process;
```

CallHistory.tla v1->v2

```
1 | SPECIFICATION Spec
2 | /* Add statements after this line.
3 |
4 | CONSTANTS
5 |   Calls = {"call-1", "call-2", "call-3"}
6 |
7 | PROPERTIES
8 |   Success
9 |   Termination
```

CallHistory.tla v2 Run: Failed

```
21 | Error: Temporal properties were violated.  
22 |  
23 | Error: The following behavior constitutes a counter-example:  
306 |  
307 | State 19: <Process line 120, col 12 to line 126, col 36 of module CallHistory>  
308 | /\ store = <<"call-3", "call-2">>  
309 | /\ wip = ("call-1" :> [offset |-> 5])  
310 | /\ pc = ( "call-1" :> "Done" &&  
311 |   "call-2" :> "Done" &&  
312 |   "call-3" :> "Done" &&  
313 |   "Shutdown" :> "Done" &&  
314 |   "Processor" :> "Done" )  
315 | /\ offset = 6  
316 | /\ shutdownProcessor = TRUE  
317 | /\ queue = << [id |-> "call-2", action |-> "setup"],  
318 |   [id |-> "call-3", action |-> "setup"],  
319 |   [id |-> "call-3", action |-> "complete"],  
320 |   [id |-> "call-2", action |-> "complete"],  
321 |   [id |-> "call-1", action |-> "setup"],  
322 |   [id |-> "call-1", action |-> "complete"] >>  
323 | /\ committed = 5
```

CallHistory.tla v2 Run: Failed

```
21 | Error: Temporal properties were violated.  
22 |  
23 | Error: The following behavior constitutes a counter-example:  
306 |  
307 | State 19: <Process line 120, col 12 to line 126, col 36 of module CallHistory>  
308 | /\ store = <<"call-3", "call-2">>  
309 | /\ wip = ("call-1" :> [offset |-> 5])  
310 | /\ pc = ( "call-1" :> "Done" &&  
311 |   "call-2" :> "Done" &&  
312 |   "call-3" :> "Done" &&  
313 |   "Shutdown" :> "Done" &&  
314 |   "Processor" :> "Done" )  
315 | /\ offset = 6  
316 | /\ shutdownProcessor = TRUE  
317 | /\ queue = << [id |-> "call-2", action |-> "setup"],  
318 |   [id |-> "call-3", action |-> "setup"],  
319 |   [id |-> "call-3", action |-> "complete"],  
320 |   [id |-> "call-2", action |-> "complete"],  
321 |   [id |-> "call-1", action |-> "setup"],  
322 |   [id |-> "call-1", action |-> "complete"] >>  
323 | /\ committed = 5
```

CallHistory.tla v2 Run: Failed

```
21 | Error: Temporal properties were violated.  
22 |  
23 | Error: The following behavior constitutes a counter-example:  
306 |  
307 | State 19: <Process line 120, col 12 to line 126, col 36 of module CallHistory>  
308 | /\ store = <<"call-3", "call-2">>  
309 | /\ wip = ("call-1" :> [offset |-> 5])  
310 | /\ pc = ( "call-1" :> "Done"   
311 |   "call-2" :> "Done"   
312 |   "call-3" :> "Done"   
313 |   "Shutdown" :> "Done"   
314 |   "Processor" :> "Done" )  
315 | /\ offset = 6  
316 | /\ shutdownProcessor = TRUE  
317 | /\ queue = << [id |-> "call-2", action |-> "setup"],  
318 |   [id |-> "call-3", action |-> "setup"],  
319 |   [id |-> "call-3", action |-> "complete"],  
320 |   [id |-> "call-2", action |-> "complete"],  
321 |   [id |-> "call-1", action |-> "setup"],  
322 |   [id |-> "call-1", action |-> "complete"] >>  
323 | /\ committed = 5
```

CallHistory.tla v2 Run: Failed

```
21 | Error: Temporal properties were violated.  
22 |  
23 | Error: The following behavior constitutes a counter-example:  
306 |  
307 | State 19: <Process line 120, col 12 to line 126, col 36 of module CallHistory>  
308 | /\ store = <<"call-3", "call-2">>  
309 | /\ wip = ("call-1" :> [offset |-> 5])  
310 | /\ pc = ( "call-1" :> "Done" /\  
311 |   "call-2" :> "Done" /\  
312 |   "call-3" :> "Done" /\  
313 |   "Shutdown" :> "Done" /\  
314 |   "Processor" :> "Done" )  
315 | /\ offset = 6  
316 | /\ shutdownProcessor = TRUE  
317 | /\ queue = << [id |-> "call-2", action |-> "setup"],  
318 |   [id |-> "call-3", action |-> "setup"],  
319 |   [id |-> "call-3", action |-> "complete"],  
320 |   [id |-> "call-2", action |-> "complete"],  
321 |   [id |-> "call-1", action |-> "setup"],  
322 |   [id |-> "call-1", action |-> "complete"] >>  
323 | /\ committed = 5
```

CallHistory.tla v2->v3

```
50 begin Process:  
51     while TRUE do  
52         either  
53             when Len(queue) >= offset;  
54             ProcessCDR:  
55                 with cdr = queue[offset]  
56                 do  
57                     if cdr.id \notin DOMAIN wip /\ cdr.action = "setup" then  
58                         wip := Add(wip, cdr.id, [offset |-> offset]);  
59                     elsif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then  
60                         wip := Remove(wip, cdr.id);  
61                         store := Append(store, cdr.id);  
62                     end if;  
63                 end with;  
64                 committed := offset;  
65                 offset := offset + 1;  
66             or  
67             -         when shutdownProcessor;  
68             +         when shutdownProcessor /\ wip = <>>;  
69                 goto Done;  
70             end either;  
71         end while;
```

CallHistory.tla v3 Run: Failed

```
21 | Error: Temporal properties were violated.  
22 |  
23 | Error: The following behavior constitutes a counter-example:  
24 |  
148 |  
149 | State 9: <Process line 120, col 12 to line 126, col 36 of module CallHistory>  
150 | /\ store = <>>  
151 | /\ wip = <>>  
152 | /\ pc = ( "call-1" :> "Done" /\  
153 |   "call-2" :> "Done" /\  
154 |   "call-3" :> "Done" /\  
155 |   "Shutdown" :> "Done" /\  
156 |   "Processor" :> "Done" )  
157 | /\ offset = 1  
158 | /\ shutdownProcessor = TRUE  
159 | /\ queue = << [id |-> "call-2", action |-> "setup"],  
160 |   [id |-> "call-2", action |-> "complete"],  
161 |   [id |-> "call-1", action |-> "setup"],  
162 |   [id |-> "call-1", action |-> "complete"],  
163 |   [id |-> "call-3", action |-> "setup"],  
164 |   [id |-> "call-3", action |-> "complete"] >>  
165 | /\ committed = 0
```

CallHistory.tla v3 Run: Failed

```
21 | Error: Temporal properties were violated.  
22 |  
23 | Error: The following behavior constitutes a counter-example:  
24 |  
148 |  
149 | State 9: <Process line 120, col 12 to line 126, col 36 of module CallHistory>  
150 | /\ store = <>>  
151 | /\ wip = <>>  
152 | /\ pc = ( "call-1" :> "Done" /\  
153 |   "call-2" :> "Done" /\  
154 |   "call-3" :> "Done" /\  
155 |   "Shutdown" :> "Done" /\  
156 |   "Processor" :> "Done" )  
157 | /\ offset = 1  
158 | /\ shutdownProcessor = TRUE  
159 | /\ queue = << [id |-> "call-2", action |-> "setup"],  
160 |   [id |-> "call-2", action |-> "complete"],  
161 |   [id |-> "call-1", action |-> "setup"],  
162 |   [id |-> "call-1", action |-> "complete"],  
163 |   [id |-> "call-3", action |-> "setup"],  
164 |   [id |-> "call-3", action |-> "complete"] >>  
165 | /\ committed = 0
```

CallHistory.tla v3 Run: Failed

```
21 | Error: Temporal properties were violated.  
22 |  
23 | Error: The following behavior constitutes a counter-example:  
24 |  
148 |  
149 | State 9: <Process line 120, col 12 to line 126, col 36 of module CallHistory>  
150 | /\ store = <>>  
151 | /\ wip = <>>  
152 | /\ pc = ( "call-1" :> "Done" ;&  
153 |   "call-2" :> "Done" ;&  
154 |   "call-3" :> "Done" ;&  
155 |   "Shutdown" :> "Done" ;&  
156 |   "Processor" :> "Done" )  
157 | /\ offset = 1  
158 | /\ shutdownProcessor = TRUE  
159 | /\ queue = << [id |-> "call-2", action |-> "setup"],  
160 |   [id |-> "call-2", action |-> "complete"],  
161 |   [id |-> "call-1", action |-> "setup"],  
162 |   [id |-> "call-1", action |-> "complete"],  
163 |   [id |-> "call-3", action |-> "setup"],  
164 |   [id |-> "call-3", action |-> "complete"] >>  
165 | /\ committed = 0
```

CallHistory.tla v3 Run: Failed

```
21 | Error: Temporal properties were violated.  
22 |  
23 | Error: The following behavior constitutes a counter-example:  
24 |  
148 |  
149 State 9: <Process line 120, col 12 to line 126, col 36 of module CallHistory>  
150 | \ store = <>>  
151 | \ wip = <>>  
152 | \ pc = ( "call-1" :> "Done" ;&  
153 |   "call-2" :> "Done" ;&  
154 |   "call-3" :> "Done" ;&  
155 |   "Shutdown" :> "Done" ;&  
156 |   "Processor" :> "Done" )  
157 | \ offset = 1  
158 | \ shutdownProcessor = TRUE  
159 | \ queue = << [id |-> "call-2", action |-> "setup"],  
160 |   [id |-> "call-2", action |-> "complete"],  
161 |   [id |-> "call-1", action |-> "setup"],  
162 |   [id |-> "call-1", action |-> "complete"],  
163 |   [id |-> "call-3", action |-> "setup"],  
164 |   [id |-> "call-3", action |-> "complete"] >>  
165 | \ committed = 0
```

CallHistory.tla v3->v4

```
54 ProcessCDR:  
55     with cdr = queue[offset]  
56     do  
57         if cdr.id \notin DOMAIN wip \wedge cdr.action = "setup" then  
58             wip := Add(wip, cdr.id, [offset |-> offset]);  
59         elsif cdr.id \in DOMAIN wip \wedge cdr.action = "complete" then  
60             wip := Remove(wip, cdr.id);  
61             store := Append(store, cdr.id);  
62         end if;  
63     end with;  
64     committed := offset;  
65     offset := offset + 1;  
66     or  
67     -     when shutdownProcessor \wedge wip = <>>;  
68     +     when shutdownProcessor \wedge wip = <>> \wedge offset > Len(queue);  
69         goto Done;  
70     end either;  
71     end while;  
72 end process;
```

CallHistory.tla v4 Run: Succeed

```
22 Model checking completed. No error has been found.  
23 Estimates of the probability that TLC did not check all reachable states  
24 because two distinct states had the same fingerprint:  
25 calculated (optimistic): val = 7.2E-13  
26 7357 states generated, 4129 distinct states found, 0 states left on queue.  
27 The depth of the complete state graph search is 21.
```

CallHistory.tla v4 Run: Succeed

```
22 Model checking completed. No error has been found.  
23 Estimates of the probability that TLC did not check all reachable states  
24 because two distinct states had the same fingerprint:  
25 calculated (optimistic): val = 7.2E-13  
26 7357 states generated, 4129 distinct states found, 0 states left on queue.  
27 The depth of the complete state graph search is 21.
```

**Модель:
добавим аварийный останов**

CallHistory.tla v4->v5

```
47  fair process Processor = "Processor"
48  variables
49      wip = <<>>,
50 +    crashes = 0,
51     offset = committed + 1;
52 begin Process:
53     while TRUE do
54         either
55             when Len(queue) >= offset;
56 ProcessCDR:
57
58 +         or
59 +         when crashes < Crashes;
60 +Crash:
61     +         crashes := crashes + 1 ||
62     +         offset := committed + 1 ||
63     +         wip := <<>>;
```

CallHistory.tla v4->v5

```
47  fair process Processor = "Processor"
48  variables
49      wip = <<>>,
50 +   crashes = 0,
51     offset = committed + 1;
52 begin Process:
53     while TRUE do
54         either
55             when Len(queue) >= offset;
56 ProcessCDR:
57
58 +         or
59 +         when crashes < Crashes;
60 +Crash:
61 +         crashes := crashes + 1 ||
62 +         offset := committed + 1 ||
63 +         wip := <<>>;
```

CallHistory.tla v4->v5

```
1 SPECIFICATION Spec
2 /* Add statements after this line.
3
4 CONSTANTS
5   Calls = {"call-1", "call-2", "call-3"}
6   Crashes = 1
7
8 PROPERTIES
9   Success
10  Termination
```

CallHistory.tla v5 Run: Failed

```
21 Error: Temporal properties were violated.  
22  
23 Error: The following behavior constitutes a counter-example:  
405  
406 State 23: <Process line 130, col 12 to line 138, col 45 of module CallHistory>  
407 /\ store = <<"call-2", "call-3">>  
408 /\ wip = << >>  
409 /\ crashes = 1  
410 /\ pc = ( "call-1" :> "Done" ;&  
411   "call-2" :> "Done" ;&  
412   "call-3" :> "Done" ;&  
413   "Shutdown" :> "Done" ;&  
414   "Processor" :> "Done" )  
415 /\ offset = 7  
416 /\ shutdownProcessor = TRUE  
417 /\ queue = << [id |-> "call-1", action |-> "setup"],  
418   [id |-> "call-2", action |-> "setup"],  
419   [id |-> "call-3", action |-> "setup"],  
420   [id |-> "call-1", action |-> "complete"],  
421   [id |-> "call-2", action |-> "complete"],  
422   [id |-> "call-3", action |-> "complete"] >>  
423 /\ committed = 6  
424  
425 State 24: Stuttering
```

CallHistory.tla v5 Run: Failed

```
21 | Error: Temporal properties were violated.  
22 |  
23 | Error: The following behavior constitutes a counter-example:  
405 |  
406 | State 23: <Process line 130, col 12 to line 138, col 45 of module CallHistory>  
407 | /\ store = <<"call-2", "call-3">>  
408 | /\ wip = << >>  
409 | /\ crashes = 1  
410 | /\ pc = ( "call-1" :> "Done" /\  
411 |   "call-2" :> "Done" /\  
412 |   "call-3" :> "Done" /\  
413 |   "Shutdown" :> "Done" /\  
414 |   "Processor" :> "Done" )  
415 | /\ offset = 7  
416 | /\ shutdownProcessor = TRUE  
417 | /\ queue = << [id |-> "call-1", action |-> "setup"],  
418 |   [id |-> "call-2", action |-> "setup"],  
419 |   [id |-> "call-3", action |-> "setup"],  
420 |   [id |-> "call-1", action |-> "complete"],  
421 |   [id |-> "call-2", action |-> "complete"],  
422 |   [id |-> "call-3", action |-> "complete"] >>  
423 | /\ committed = 6  
424 |  
425 | State 24: Stuttering
```

CallHistory.tla v5 Run: Failed

```
21 | Error: Temporal properties were violated.  
22 |  
23 | Error: The following behavior constitutes a counter-example:  
405 |  
406 | State 23: <Process line 130, col 12 to line 138, col 45 of module CallHistory>  
407 | /\ store = <<"call-2", "call-3">>  
408 | /\ wip = << >>  
409 | /\ crashes = 1  
410 | /\ pc = ( "call-1" :> "Done" ;&  
411 |   "call-2" :> "Done" ;&  
412 |   "call-3" :> "Done" ;&  
413 |   "Shutdown" :> "Done" ;&  
414 |   "Processor" :> "Done" )  
415 | /\ offset = 7  
416 | /\ shutdownProcessor = TRUE  
417 | /\ queue = << [id |-> "call-1", action |-> "setup"],  
418 |   [id |-> "call-2", action |-> "setup"],  
419 |   [id |-> "call-3", action |-> "setup"],  
420 |   [id |-> "call-1", action |-> "complete"],  
421 |   [id |-> "call-2", action |-> "complete"],  
422 |   [id |-> "call-3", action |-> "complete"] >>  
423 | /\ committed = 6  
424 |  
425 | State 24: Stuttering
```

CallHistory.tla v5 Run: Failed

```
21 Error: Temporal properties were violated.  
22  
23 Error: The following behavior constitutes a counter-example:  
405  
406 State 23: <Process line 130, col 12 to line 138, col 45 of module CallHistory>  
407 /\ store = <<"call-2", "call-3">>  
408 /\ wip = << >>  
409 /\ crashes = 1  
410 /\ pc = ( "call-1" :> "Done" ;&  
411   "call-2" :> "Done" ;&  
412   "call-3" :> "Done" ;&  
413   "Shutdown" :> "Done" ;&  
414   "Processor" :> "Done" )  
415 /\ offset = 7  
416 /\ shutdownProcessor = TRUE  
417 /\ queue = << [id |-> "call-1", action |-> "setup"],  
418   [id |-> "call-2", action |-> "setup"],  
419   [id |-> "call-3", action |-> "setup"],  
420   [id |-> "call-1", action |-> "complete"],  
421   [id |-> "call-2", action |-> "complete"],  
422   [id |-> "call-3", action |-> "complete"] >>  
423 /\ committed = 6  
424  
425 State 24: Stuttering
```

CallHistory.tla v5 Run: Failed

```
21 | Error: Temporal properties were violated.  
22 |  
23 | Error: The following behavior constitutes a counter-example:  
405 |  
406 | State 23: <Process line 130, col 12 to line 138, col 45 of module CallHistory>  
407 | /\ store = <<"call-2", "call-3">>  
408 | /\ wip = << >>  
409 | /\ crashes = 1  
410 | /\ pc = ( "call-1" :> "Done" /\&  
411 |   "call-2" :> "Done" /\&  
412 |   "call-3" :> "Done" /\&  
413 |   "Shutdown" :> "Done" /\&  
414 |   "Processor" :> "Done" )  
415 | /\ offset = 7  
416 | /\ shutdownProcessor = TRUE  
417 | /\ queue = << [id |-> "call-1", action |-> "setup"],  
418 |   [id |-> "call-2", action |-> "setup"],  
419 |   [id |-> "call-3", action |-> "setup"],  
420 |   [id |-> "call-1", action |-> "complete"],  
421 |   [id |-> "call-2", action |-> "complete"],  
422 |   [id |-> "call-3", action |-> "complete"] >>  
423 | /\ committed = 6  
424 |  
425 | State 24: Stuttering
```

CallHistory.tla v5 Run: Failed

```
142 | State 9: <CallComplete line 114, col 23 to line 118, col 56 of module CallHistory>
143 | /\ store = <>>
144 | /\ wip = ("call-1" :> [offset |-> 1])
145 | /\ crashes = 0
146 | /\ pc = ( "call-1" :> "Done" /\&
147 |   "call-2" :> "Done" /\&
148 |   "call-3" :> "CallComplete" /\&
149 |   "Shutdown" :> "SignalShutdown" /\&
150 |   "Processor" :> "Crash" )
151 | /\ offset = 2
152 | /\ shutdownProcessor = FALSE
153 | /\ queue = << [id |-> "call-1", action |-> "setup"],
154 |   [id |-> "call-2", action |-> "setup"],
155 |   [id |-> "call-3", action |-> "setup"],
156 |   [id |-> "call-1", action |-> "complete"],
157 |   [id |-> "call-2", action |-> "complete"] >>
158 | /\ committed = 1
```

CallHistory.tla v5 Run: Failed

```
142 State 9: <CallComplete line 114, col 23 to line 118, col 56 of module CallHistory>
143 /\ store = <>>
144 /\ wip = ("call-1" :> [offset |-> 1])
145 /\ crashes = 0
146 /\ pc = ( "call-1" :> "Done" &&
147   "call-2" :> "Done" &&
148   "call-3" :> "CallComplete" &&
149   "Shutdown" :> "SignalShutdown" &&
150   "Processor" :> "Crash" )
151 /\ offset = 2
152 /\ shutdownProcessor = FALSE
153 /\ queue = << [id |-> "call-1", action |-> "setup"],
154   [id |-> "call-2", action |-> "setup"],
155   [id |-> "call-3", action |-> "setup"],
156   [id |-> "call-1", action |-> "complete"],
157   [id |-> "call-2", action |-> "complete"] >>
158 /\ committed = 1
```

CallHistory.tla v5 Run: Failed

```
142 State 9: <CallComplete line 114, col 23 to line 118, col 56 of module CallHistory>
143 /\ store = <>>
144 /\ wip = ("call-1" :> [offset |-> 1])
145 /\ crashes = 0
146 /\ pc = ( "call-1" :> "Done" &&
147   "call-2" :> "Done" &&
148   "call-3" :> "CallComplete" &&
149   "Shutdown" :> "SignalShutdown" &&
150   "Processor" :> "Crash" )
151 /\ offset = 2
152 /\ shutdownProcessor = FALSE
153 /\ queue = << [id |-> "call-1", action |-> "setup"],
154   [id |-> "call-2", action |-> "setup"],
155   [id |-> "call-3", action |-> "setup"],
156   [id |-> "call-1", action |-> "complete"],
157   [id |-> "call-2", action |-> "complete"] >>
158 /\ committed = 1
```

CallHistory.tla v5 Run: Failed

```
160 State 10: <Crash line 155, col 10 to line 160, col 69 of module CallHistory>
161 /\ store = <>>
162 /\ wip = <>>
163 /\ crashes = 1
164 /\ pc = ( "call-1" :> "Done" ;;
165   "call-2" :> "Done" ;;
166   "call-3" :> "CallComplete" ;;
167   "Shutdown" :> "SignalShutdown" ;;
168   "Processor" :> "Process" )
169 /\ offset = 2
170 /\ shutdownProcessor = FALSE
171 /\ queue = << [id |-> "call-1", action |-> "setup"],
172   [id |-> "call-2", action |-> "setup"],
173   [id |-> "call-3", action |-> "setup"],
174   [id |-> "call-1", action |-> "complete"],
175   [id |-> "call-2", action |-> "complete"] >>
176 /\ committed = 1
```

CallHistory.tla v5->v6

```
13  | +set ++ x == set \union {x}
58  |   fair process Processor = "Processor"
59  |     variables
60  |       wip = <>>,
61  | +    processed = {},
62  |     crashes = 0,
63  |     offset = committed + 1;
64  begin Process:
65    while TRUE do
66      either
67        when Len(queue) >= offset;
68      ProcessCDR:
69        with cdr = queue[offset]
70        do
71          if cdr.id \notin DOMAIN wip /\ cdr.action = "setup" then
72            wip := Add(wip, cdr.id, [offset |-> offset]);
73          elsif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then
74 +            processed := processed ++ wip[cdr.id].offset ++ offset;
75            wip := Remove(wip, cdr.id);
76 +            committed := CommitOffset(processed, committed);
77            store := Append(store, cdr.id);
78          end if;
79        end with;
80 -        committed := offset;
```

CallHistory.tla v5->v6

```
13  | +set ++ x == set \union {x}
58  |   fair process Processor = "Processor"
59  |     variables
60  |       wip = <>>,
61  | +     processed = {},
62  |     crashes = 0,
63  |     offset = committed + 1;
64  begin Process:
65    while TRUE do
66      either
67        when Len(queue) >= offset;
68      ProcessCDR:
69        with cdr = queue[offset]
70        do
71          if cdr.id \notin DOMAIN wip /\ cdr.action = "setup" then
72            wip := Add(wip, cdr.id, [offset |-> offset]);
73          elsif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then
74 +            processed := processed ++ wip[cdr.id].offset ++ offset;
75            wip := Remove(wip, cdr.id);
76 +            committed := CommitOffset(processed, committed);
77            store := Append(store, cdr.id);
78          end if;
79        end with;
80 -        committed := offset;
```

CallHistory.tla v5->v6

```
13  | +set ++ x == set \union {x}
58  |   fair process Processor = "Processor"
59  |     variables
60  |       wip = <>>,
61  | +     processed = {},
62  |     crashes = 0,
63  |     offset = committed + 1;
64  begin Process:
65    while TRUE do
66      either
67        when Len(queue) >= offset;
68      ProcessCDR:
69        with cdr = queue[offset]
70        do
71          if cdr.id \notin DOMAIN wip /\ cdr.action = "setup" then
72            wip := Add(wip, cdr.id, [offset |-> offset]);
73          elsif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then
74 +            processed := processed ++ wip[cdr.id].offset ++ offset;
75            wip := Remove(wip, cdr.id);
76 +            committed := CommitOffset(processed, committed);
77            store := Append(store, cdr.id);
78            end if;
79        end with;
80 -        committed := offset;
```

CallHistory.tla v5->v6

```
81          offset := offset + 1;
82      or
83          when shutdownProcessor /\ wip = <>> /\ offset > Len(queue);
84              goto Done;
85      or
86          when crashes < Crashes;
87 Crash:
88         crashes := crashes + 1 ||
89 +         processed := {} ||
90         offset := committed + 1 ||
91         wip := <>>;
92     end either;
93 end while;
94 end process;
```

CallHistory.tla v5->v6

```
24  define
25
26 +    \* {1, 3}, 0 -> 0
27 +    \* {2, 3}, 0 -> 0
28 +    \* {1, 2, 3}, 0 -> 3
```

CallHistory.tla v5->v6

```
24 define
25
26 +   \* {1, 3}, 0 -> 0
27 +   \* {2, 3}, 0 -> 0
28 +   \* {1, 2, 3}, 0 -> 3
29 +   CommitOffset(set, current) ==
30 +     LET first == current + 1 IN
31 +       IF /\ first \in set
32 +         /\ \A i \in {j \in set: j > first}: i-1 \in set
33 +         THEN CHOOSE i \in set: i+1 \notin set
34 +       ELSE current
35 +
36     AllCallsProcessed == <>[ ](
37       \A i \in Calls:
38         Len(SelectSeq(store, LAMBDA j: j = i)) = 1
39     )
40
41     Success == AllCallsProcessed
42
43 end define;
```

CallHistory.tla v5->v6

```
24 define
25
26 +   \* {1, 3}, 0 -> 0
27 +   \* {2, 3}, 0 -> 0
28 +   \* {1, 2, 3}, 0 -> 3
29 +   CommitOffset(set, current) ==
30 +     LET first == current + 1 IN
31 +     IF /\ first \in set
32 +       /\ \A i \in {j \in set: j > first}: i-1 \in set
33 +       THEN CHOOSE i \in set: i+1 \notin set
34 +     ELSE current
35 +
36     AllCallsProcessed == <>[ ](
37       \A i \in Calls:
38         Len(SelectSeq(store, LAMBDA j: j = i)) = 1
39     )
40
41     Success == AllCallsProcessed
42
43 end define;
```

CallHistory.tla v5->v6

```
24 define
25
26 +   \* {1, 3}, 0 -> 0
27 +   \* {2, 3}, 0 -> 0
28 +   \* {1, 2, 3}, 0 -> 3
29 +   CommitOffset(set, current) ==
30 +     LET first == current + 1 IN
31 +     IF /\ first \in set
32 +       /\ \A i \in {j \in set: j > first}: i-1 \in set
33 +       THEN CHOOSE i \in set: i+1 \notin set
34 +     ELSE current
35 +
36     AllCallsProcessed == <>[ ](
37       \A i \in Calls:
38         Len(SelectSeq(store, LAMBDA j: j = i)) = 1
39     )
40
41     Success == AllCallsProcessed
42
43 end define;
```

CallHistory.tla v5->v6

```
24 define
25
26 +   \* {1, 3}, 0 -> 0
27 +   \* {2, 3}, 0 -> 0
28 +   \* {1, 2, 3}, 0 -> 3
29 +   CommitOffset(set, current) ==
30 +     LET first == current + 1 IN
31 +     IF /\ first \in set
32 +       /\ \A i \in {j \in set: j > first}: i-1 \in set
33 +       THEN CHOOSE i \in set: i+1 \notin set
34 +     ELSE current
35 +
36     AllCallsProcessed == <>[ ](
37       \A i \in Calls:
38         Len(SelectSeq(store, LAMBDA j: j = i)) = 1
39     )
40
41     Success == AllCallsProcessed
42
43 end define;
```

CallHistory.tla v5->v6

```
24 define
25
26 +   \* {1, 3}, 0 -> 0
27 +   \* {2, 3}, 0 -> 0
28 +   \* {1, 2, 3}, 0 -> 3
29 +   CommitOffset(set, current) ==
30 +     LET first == current + 1 IN
31 +     IF /\ first \in set
32 +       /\ \A i \in {j \in set: j > first}: i-1 \in set
33 +       THEN CHOOSE i \in set: i+1 \notin set
34 +     ELSE current
35 +
36     AllCallsProcessed == <>[ ](
37       \A i \in Calls:
38         Len(SelectSeq(store, LAMBDA j: j = i)) = 1
39     )
40
41     Success == AllCallsProcessed
42
43 end define;
```

CallHistory.tla v6 Run: Failed

```
23 Error: The following behavior constitutes a counter-example:  
542  
543 State 29: <Process line 153, col 12 to line 161, col 56 of module CallHistory>  
544 /\ store = <<"call-1", "call-1", "call-2", "call-3">>  
545 /\ processed = {1, 2, 3, 4, 5, 6}  
546 /\ wip = << >>  
547 /\ crashes = 1  
548 /\ pc = ( "call-1" :> "Done" @@  
549   "call-2" :> "Done" @@  
550   "call-3" :> "Done" @@  
551   "Shutdown" :> "Done" @@  
552   "Processor" :> "Done" )  
553 /\ offset = 7  
554 /\ shutdownProcessor = TRUE  
555 /\ queue = << [id |-> "call-2", action |-> "setup"],  
556   [id |-> "call-1", action |-> "setup"],  
557   [id |-> "call-1", action |-> "complete"],  
558   [id |-> "call-2", action |-> "complete"],  
559   [id |-> "call-3", action |-> "setup"],  
560   [id |-> "call-3", action |-> "complete"] >>  
561 /\ committed = 6  
562  
563 State 30: Stuttering
```

CallHistory.tla v6 Run: Failed

```
23 | Error: The following behavior constitutes a counter-example:  
542 |  
543 | State 29: <Process line 153, col 12 to line 161, col 56 of module CallHistory>  
544 | /\ store = <<"call-1", "call-1", "call-2", "call-3">>  
545 | /\ processed = {1, 2, 3, 4, 5, 6}  
546 | /\ wip = << >>  
547 | /\ crashes = 1  
548 | /\ pc = ( "call-1" :> "Done" ;&  
549 |   "call-2" :> "Done" ;&  
550 |   "call-3" :> "Done" ;&  
551 |   "Shutdown" :> "Done" ;&  
552 |   "Processor" :> "Done" )  
553 | /\ offset = 7  
554 | /\ shutdownProcessor = TRUE  
555 | /\ queue = << [id |-> "call-2", action |-> "setup"],  
556 |   [id |-> "call-1", action |-> "setup"],  
557 |   [id |-> "call-1", action |-> "complete"],  
558 |   [id |-> "call-2", action |-> "complete"],  
559 |   [id |-> "call-3", action |-> "setup"],  
560 |   [id |-> "call-3", action |-> "complete"] >>  
561 | /\ committed = 6  
562 |  
563 | State 30: Stuttering
```

CallHistory.tla v6 Run: Failed

```
23 | Error: The following behavior constitutes a counter-example:  
542 |  
543 | State 29: <Process line 153, col 12 to line 161, col 56 of module CallHistory>  
544 | /\ store = <<"call-1", "call-1", "call-2", "call-3">>  
545 | /\ processed = {1, 2, 3, 4, 5, 6}  
546 | /\ wip = << >>  
547 | /\ crashes = 1  
548 | /\ pc = ( "call-1" :> "Done" ;&  
549 |   "call-2" :> "Done" ;&  
550 |   "call-3" :> "Done" ;&  
551 |   "Shutdown" :> "Done" ;&  
552 |   "Processor" :> "Done" )  
553 | /\ offset = 7  
554 | /\ shutdownProcessor = TRUE  
555 | /\ queue = << [id |-> "call-2", action |-> "setup"],  
556 |   [id |-> "call-1", action |-> "setup"],  
557 |   [id |-> "call-1", action |-> "complete"],  
558 |   [id |-> "call-2", action |-> "complete"],  
559 |   [id |-> "call-3", action |-> "setup"],  
560 |   [id |-> "call-3", action |-> "complete"] >>  
561 | /\ committed = 6  
562 |  
563 | State 30: Stuttering
```

CallHistory.tla v6->v7

```
19 | variables
20 |     queue = <>>,
21 |     committed = 0,
22 |     shutdownProcessor = FALSE,
23 | -     store = <>>;
24 | +     store = {};
25 |
26 |
27 | -     AllCallsProcessed == <>[()
28 | -         \A i \in Calls:
29 | -             Len(SelectSeq(store, LAMBDA j: j = i)) = 1
30 | -     )
31 | +     AllCallsProcessed == <>[()Calls = store]
32 |
33 |
34 |         elseif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then
35 |             processed := processed ++ wip[cdr.id].offset ++ offset;
36 |             wip := Remove(wip, cdr.id);
37 |             committed := CommitOffset(processed, committed);
38 | -             store := Append(store, cdr.id);
39 | +             store := store ++ cdr.id;
40 |         end if;
```

CallHistory.tla v6->v7

```
19 | variables
20 |     queue = <>>,
21 |     committed = 0,
22 |     shutdownProcessor = FALSE,
23 | -     store = <>>;
24 | +     store = {};
36 |
37 | -     AllCallsProcessed == <>[()
38 | -         \A i \in Calls:
39 | -             Len(SelectSeq(store, LAMBDA j: j = i)) = 1
40 | -     )
41 | +     AllCallsProcessed == <>[()Calls = store]
42 |
75 |             elseif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then
76 |                 processed := processed ++ wip[cdr.id].offset ++ offset;
77 |                 wip := Remove(wip, cdr.id);
78 |                 committed := CommitOffset(processed, committed);
79 | -                 store := Append(store, cdr.id);
80 | +                     store := store ++ cdr.id;
81 |             end if;
```

CallHistory.tla v6->v7

```
19 | variables
20 |     queue = <>>,
21 |     committed = 0,
22 |     shutdownProcessor = FALSE,
23 | -     store = <>>;
24 | +     store = {};
25 |
26 |
27 | -     AllCallsProcessed == <>[|(
28 | -         \A i \in Calls:
29 | -             Len(SelectSeq(store, LAMBDA j: j = i)) = 1
30 | -     )
31 | +     AllCallsProcessed == <>[|(Calls = store)
32 |
33 |
34 |
35 |
36 |
37 |
38 |
39 |
40 |
41 |
42 |
43 |
44 |
45 |
46 |
47 |
48 |
49 |
50 |
51 |
52 |
53 |
54 |
55 |
56 |
57 |
58 |
59 |
60 |
61 |
62 |
63 |
64 |
65 |
66 |
67 |
68 |
69 |
70 |
71 |
72 |
73 |
74 |
75 |         elseif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then
76 |             processed := processed ++ wip[cdr.id].offset ++ offset;
77 |             wip := Remove(wip, cdr.id);
78 |             committed := CommitOffset(processed, committed);
79 | -             store := Append(store, cdr.id);
80 | +             store := store ++ cdr.id;
81 |         end if;
```

CallHistory.tla v7 Run: Succeed

```
22 Model checking completed. No error has been found.  
23 Estimates of the probability that TLC did not check all reachable states  
24 because two distinct states had the same fingerprint:  
25 calculated (optimistic): val = 1.2E-11  
26 29490 states generated, 15435 distinct states found, 0 states left on queue.  
27 The depth of the complete state graph search is 29.
```

CallHistory.tla v7 Run: Succeed

```
22 Model checking completed. No error has been found.  
23 Estimates of the probability that TLC did not check all reachable states  
24 because two distinct states had the same fingerprint:  
25 calculated (optimistic): val = 1.2E-11  
26 29490 states generated, 15435 distinct states found, 0 states left on queue.  
27 The depth of the complete state graph search is 29.
```

**Модель:
добавим аварийный сброс вызова**

CallHistory.tla v7->v8

```
42 fair process Call \in Calls
43 begin CallSetup:
44     queue := Append(queue, [id |-> self, action |-> "setup"]);
45     CallComplete:
46     + either
47         queue := Append(queue, [id |-> self, action |-> "complete"]);
48     + or
49     + skip;
50     + end either;
51 end process;
```

CallHistory.tla v7->v8

```
42 fair process Call \in Calls
43 begin CallSetup:
44     queue := Append(queue, [id |-> self, action |-> "setup"]);
45     CallComplete:
46     + either
47         queue := Append(queue, [id |-> self, action |-> "complete"]);
48     + or
49     + skip;
50     + end either;
51 end process;
```

CallHistory.tla v8 Run: Failed

```
21 | Error: Temporal properties were violated.  
22 |  
23 | Error: The following behavior constitutes a counter-example:  
356 |  
357 State 20: <ProcessCDR line 163, col 15 to line 178, col 66 of module CallHistory>  
358 /\ store = {"call-1", "call-3"}  
359 /\ processed = {1, 2, 3, 4}  
360 /\ wip = ("call-2" :> [offset |-> 5])  
361 /\ crashes = 1  
362 /\ pc = ( "call-1" :> "Done" Ȧ  
363     "call-2" :> "Done" Ȧ  
364     "call-3" :> "Done" Ȧ  
365     "Shutdown" :> "Done" Ȧ  
366     "Processor" :> "Process" )  
367 /\ offset = 6  
368 /\ shutdownProcessor = TRUE  
369 /\ queue = << [id |-> "call-3", action |-> "setup"],  
370     [id |-> "call-3", action |-> "complete"],  
371     [id |-> "call-1", action |-> "setup"],  
372     [id |-> "call-1", action |-> "complete"],  
373     [id |-> "call-2", action |-> "setup"] >>  
374 /\ committed = 4  
375 |  
376 State 21: Stuttering
```

CallHistory.tla v8 Run: Failed

```
21 | Error: Temporal properties were violated.  
22 |  
23 | Error: The following behavior constitutes a counter-example:  
356 |  
357 State 20: <ProcessCDR line 163, col 15 to line 178, col 66 of module CallHistory>  
358 /\ store = {"call-1", "call-3"}  
359 /\ processed = {1, 2, 3, 4}  
360 /\ wip = ("call-2" :> [offset |-> 5])  
361 /\ crashes = 1  
362 /\ pc = ( "call-1" :> "Done" ȏ  
363     "call-2" :> "Done" ȏ  
364     "call-3" :> "Done" ȏ  
365     "Shutdown" :> "Done" ȏ  
366     "Processor" :> "Process" )  
367 /\ offset = 6  
368 /\ shutdownProcessor = TRUE  
369 /\ queue = << [id |-> "call-3", action |-> "setup"],  
370     [id |-> "call-3", action |-> "complete"],  
371     [id |-> "call-1", action |-> "setup"],  
372     [id |-> "call-1", action |-> "complete"],  
373     [id |-> "call-2", action |-> "setup"] >>  
374 /\ committed = 4  
375 |  
376 State 21: Stuttering
```

CallHistory.tla v8 Run: Failed

```
21 | Error: Temporal properties were violated.  
22 |  
23 | Error: The following behavior constitutes a counter-example:  
356 |  
357 State 20: <ProcessCDR line 163, col 15 to line 178, col 66 of module CallHistory>  
358 /\ store = {"call-1", "call-3"}  
359 /\ processed = {1, 2, 3, 4}  
360 /\ wip = ("call-2" :> [offset |-> 5])  
361 /\ crashes = 1  
362 /\ pc = ( "call-1" :> "Done" ȏ  
363     "call-2" :> "Done" ȏ  
364     "call-3" :> "Done" ȏ  
365     "Shutdown" :> "Done" ȏ  
366     "Processor" :> "Process" )  
367 /\ offset = 6  
368 /\ shutdownProcessor = TRUE  
369 /\ queue = << [id |-> "call-3", action |-> "setup"],  
370     [id |-> "call-3", action |-> "complete"],  
371     [id |-> "call-1", action |-> "setup"],  
372     [id |-> "call-1", action |-> "complete"],  
373     [id |-> "call-2", action |-> "setup"] >>  
374 /\ committed = 4  
375 |  
376 State 21: Stuttering
```

CallHistory.tla v8->v9

```
19  variables
20      queue = <>>,
21      committed = 0,
22      shutdownProcessor = FALSE,
23 +     failed = {},
24     store = {};
37 -     AllCallsProcessed == <>[](Calls = store)
38 +     AllCallsProcessed == <>[](Calls \ failed = store)
39
40     Success == AllCallsProcessed
44 fair process Call \in Calls
45 begin CallSetup:
46     queue := Append(queue, [id |-> self, action |-> "setup"]);
47 CallComplete:
48     either
49         queue := Append(queue, [id |-> self, action |-> "complete"]);
50     or
51 -     skip;
52 +     failed := failed ++ self;
53     end either;
54 end process;
```

CallHistory.tla v8->v9

```
19  variables
20      queue = <>>,
21      committed = 0,
22      shutdownProcessor = FALSE,
23 +    failed = {},
24      store = {};
37 -    AllCallsProcessed == <>[](Calls = store)
38 +    AllCallsProcessed == <>[](Calls \ failed = store)
39
40      Success == AllCallsProcessed
44  fair process Call \in Calls
45  begin CallSetup:
46      queue := Append(queue, [id |-> self, action |-> "setup"]);
47  CallComplete:
48      either
49          queue := Append(queue, [id |-> self, action |-> "complete"]);
50      or
51 -        skip;
52 +        failed := failed ++ self;
53      end either;
54  end process;
```

CallHistory.tla v8->v9

```
19  variables
20      queue = <>>,
21      committed = 0,
22      shutdownProcessor = FALSE,
23 +    failed = {},
24      store = {};
37 -    AllCallsProcessed == <>[](Calls = store)
38 +    AllCallsProcessed == <>[](Calls \ failed = store)
39
40      Success == AllCallsProcessed
44  fair process Call \in Calls
45  begin CallSetup:
46      queue := Append(queue, [id |-> self, action |-> "setup"]);
47  CallComplete:
48      either
49          queue := Append(queue, [id |-> self, action |-> "complete"]);
50      or
51 -        skip;
52 +        failed := failed ++ self;
53      end either;
54  end process;
```

CallHistory.tla v9 Run: Failed

```
21 | Error: Temporal properties were violated.  
22 |  
23 | Error: The following behavior constitutes a counter-example:  
317 |  
318 | State 18: <ProcessCDR line 166, col 15 to line 181, col 74 of module CallHistory>  
319 | /\ store = {"call-1"}  
320 | /\ processed = {3, 4}  
321 | /\ wip = ("call-2" :> [offset |-> 1] /\ "call-3" :> [offset |-> 2])  
322 | /\ crashes = 1  
323 | /\ pc = ( "call-1" :> "Done" /\  
324 |   "call-2" :> "Done" /\  
325 |   "call-3" :> "Done" /\  
326 |   "Shutdown" :> "Done" /\  
327 |   "Processor" :> "Process" )  
328 | /\ offset = 5  
329 | /\ shutdownProcessor = TRUE  
330 | /\ queue = << [id |-> "call-2", action |-> "setup"],  
331 |   [id |-> "call-3", action |-> "setup"],  
332 |   [id |-> "call-1", action |-> "setup"],  
333 |   [id |-> "call-1", action |-> "complete"] >>  
334 | /\ committed = 0  
335 | /\ failed = {"call-2", "call-3"}  
336 |  
337 | State 19: Stuttering
```

CallHistory.tla v9 Run: Failed

```
21 | Error: Temporal properties were violated.  
22 |  
23 | Error: The following behavior constitutes a counter-example:  
317 |  
318 | State 18: <ProcessCDR line 166, col 15 to line 181, col 74 of module CallHistory>  
319 | /\ store = {"call-1"}  
320 | /\ processed = {3, 4}  
321 | /\ wip = ("call-2" :> [offset |-> 1] /\ "call-3" :> [offset |-> 2])  
322 | /\ crashes = 1  
323 | /\ pc = ( "call-1" :> "Done" /\  
324 |   "call-2" :> "Done" /\  
325 |   "call-3" :> "Done" /\  
326 |   "Shutdown" :> "Done" /\  
327 |   "Processor" :> "Process" )  
328 | /\ offset = 5  
329 | /\ shutdownProcessor = TRUE  
330 | /\ queue = << [id |-> "call-2", action |-> "setup"],  
331 |   [id |-> "call-3", action |-> "setup"],  
332 |   [id |-> "call-1", action |-> "setup"],  
333 |   [id |-> "call-1", action |-> "complete"] >>  
334 | /\ committed = 0  
335 | /\ failed = {"call-2", "call-3"}  
336 |  
337 | State 19: Stuttering
```

CallHistory.tla v9 Run: Failed

```
21 | Error: Temporal properties were violated.  
22 |  
23 | Error: The following behavior constitutes a counter-example:  
317 |  
318 | State 18: <ProcessCDR line 166, col 15 to line 181, col 74 of module CallHistory>  
319 | /\ store = {"call-1"}  
320 | /\ processed = {3, 4}  
321 | /\ wip = ("call-2" :> [offset |-> 1] /\ "call-3" :> [offset |-> 2])  
322 | /\ crashes = 1  
323 | /\ pc = ( "call-1" :> "Done" /\  
324 |   "call-2" :> "Done" /\  
325 |   "call-3" :> "Done" /\  
326 |   "Shutdown" :> "Done" /\  
327 |   "Processor" :> "Process" )  
328 | /\ offset = 5  
329 | /\ shutdownProcessor = TRUE  
330 | /\ queue = << [id |-> "call-2", action |-> "setup"],  
331 |   [id |-> "call-3", action |-> "setup"],  
332 |   [id |-> "call-1", action |-> "setup"],  
333 |   [id |-> "call-1", action |-> "complete"] >>  
334 | /\ committed = 0  
335 | /\ failed = {"call-2", "call-3"}  
336 |  
337 | State 19: Stuttering
```

CallHistory.tla v9->v10

```
10  CONSTANTS Calls
11  CONSTANTS Crashes
12 +CONSTANTS Timeout

62  fair process Processor = "Processor"
63  variables
64      wip = <<>>,
65      processed = {},
66      crashes = 0,
67 +     now = 0,
68      offset = committed + 1;
69 begin Process:
70     while TRUE do
71 +     now := now + 1;
72     either
73         when Len(queue) >= offset;
74 ProcessCDR:
75     with cdr = queue[offset]
76     do
77         if cdr.id \notin DOMAIN wip \wedge cdr.action = "setup" then
78 -             wip := Add(wip, cdr.id, [offset |-> offset]);
79 +             wip := Add(wip, cdr.id, [start |-> now, offset |-> offset]);
80         elsif cdr.id \in DOMAIN wip \wedge cdr.action = "complete" then
```

CallHistory.tla v9->v10

```
10  CONSTANTS Calls
11  CONSTANTS Crashes
12 +CONSTANTS Timeout

62  fair process Processor = "Processor"
63  variables
64      wip = <<>>,
65      processed = {},
66      crashes = 0,
67 +    now = 0,
68      offset = committed + 1;
69  begin Process:
70      while TRUE do
71 +        now := now + 1;
72          either
73              when Len(queue) >= offset;
74      ProcessCDR:
75          with cdr = queue[offset]
76          do
77              if cdr.id \notin DOMAIN wip \wedge cdr.action = "setup" then
78 -                wip := Add(wip, cdr.id, [offset |-> offset]);
79 +                wip := Add(wip, cdr.id, [start |-> now, offset |-> offset]);
80              elsif cdr.id \in DOMAIN wip \wedge cdr.action = "complete" then
```

CallHistory.tla v9->v10

```
10  CONSTANTS Calls
11  CONSTANTS Crashes
12 +CONSTANTS Timeout

62  fair process Processor = "Processor"
63  variables
64      wip = <<>>,
65      processed = {},
66      crashes = 0,
67 +    now = 0,
68      offset = committed + 1;
69  begin Process:
70      while TRUE do
71 +        now := now + 1;
72          either
73              when Len(queue) >= offset;
74      ProcessCDR:
75          with cdr = queue[offset]
76          do
77              if cdr.id \notin DOMAIN wip \wedge cdr.action = "setup" then
78 -                wip := Add(wip, cdr.id, [offset |-> offset]);
79 +                wip := Add(wip, cdr.id, [start |-> now, offset |-> offset]);
80              elsif cdr.id \in DOMAIN wip \wedge cdr.action = "complete" then
```

CallHistory.tla v9->v10

```
10  CONSTANTS Calls
11  CONSTANTS Crashes
12 +CONSTANTS Timeout

62  fair process Processor = "Processor"
63  variables
64      wip = <<>>,
65      processed = {},
66      crashes = 0,
67 +    now = 0,
68      offset = committed + 1;
69  begin Process:
70      while TRUE do
71 +        now := now + 1;
72          either
73              when Len(queue) >= offset;
74      ProcessCDR:
75          with cdr = queue[offset]
76          do
77              if cdr.id \notin DOMAIN wip \wedge cdr.action = "setup" then
78 -                wip := Add(wip, cdr.id, [offset |-> offset]);
79 +                wip := Add(wip, cdr.id, [start |-> now, offset |-> offset]);
80              elsif cdr.id \in DOMAIN wip \wedge cdr.action = "complete" then
```

CallHistory.tla v9->v10

```
47 CallComplete:
48     either
49         queue := Append(queue, [id |-> self, action |-> "complete"]);
50     or
51     -     failed := failed ++ self;
52     +     skip;
53     end either;

54     or
55         when shutdownProcessor /\ wip = <>> /\ offset > Len(queue);
56         goto Done;
57     +     or
58     +         when \E id \in DOMAIN wip: (now - wip[id].start) >= Timeout;
59 +Cleanup:
60     +     with
61     +         id = CHOOSE id \in DOMAIN wip: (now - wip[id].start) >= Timeout,
62     +         off = wip[id].offset
63     +     do
64     +         processed := processed ++ off;
65     +         wip := Remove(wip, id);
66     +         committed := CommitOffset(processed, committed);
67     +         failed := failed ++ id;
68     +     end with;
69     or
```

CallHistory.tla v9->v10

```
47 CallComplete:
48     either
49         queue := Append(queue, [id |-> self, action |-> "complete"]);
50     or
51     -     failed := failed ++ self;
52     +     skip;
53     end either;

54
55     or
56         when shutdownProcessor /\ wip = <>> /\ offset > Len(queue);
57         goto Done;
58     or
59     +     when \E id \in DOMAIN wip: (now - wip[id].start) >= Timeout;
60
61 +Cleanup:
62     with
63     +         id = CHOOSE id \in DOMAIN wip: (now - wip[id].start) >= Timeout,
64         off = wip[id].offset
65     do
66         processed := processed ++ off;
67         wip := Remove(wip, id);
68         committed := CommitOffset(processed, committed);
69         failed := failed ++ id;
70     end with;
71
72     or
```

CallHistory.tla v9->v10

```
47 CallComplete:
48     either
49         queue := Append(queue, [id |-> self, action |-> "complete"]);
50     or
51     -     failed := failed ++ self;
52     +     skip;
53     end either;

54     or
55         when shutdownProcessor /\ wip = <>> /\ offset > Len(queue);
56         goto Done;
57     or
58         when \E id \in DOMAIN wip: (now - wip[id].start) >= Timeout;
59 +Cleanup:
60     with
61         id = CHOOSE id \in DOMAIN wip: (now - wip[id].start) >= Timeout,
62         off = wip[id].offset
63     do
64         processed := processed ++ off;
65         wip := Remove(wip, id);
66         committed := CommitOffset(processed, committed);
67         failed := failed ++ id;
68     end with;
69     or
```

CallHistory.tla v9->v10

```
47 CallComplete:
48     either
49         queue := Append(queue, [id |-> self, action |-> "complete"]);
50     or
51     -     failed := failed ++ self;
52     +     skip;
53     end either;

54     or
55         when shutdownProcessor /\ wip = <>> /\ offset > Len(queue);
56         goto Done;
57     or
58         when \E id \in DOMAIN wip: (now - wip[id].start) >= Timeout;
59 +Cleanup:
60     with
61         id = CHOOSE id \in DOMAIN wip: (now - wip[id].start) >= Timeout,
62         off = wip[id].offset
63     do
64         processed := processed ++ off;
65         wip := Remove(wip, id);
66         committed := CommitOffset(processed, committed);
67         failed := failed ++ id;
68     end with;
69     or
```

CallHistory.tla v9->v10

```
47 CallComplete:
48     either
49         queue := Append(queue, [id |-> self, action |-> "complete"]);
50     or
51     -     failed := failed ++ self;
52     +     skip;
53     end either;

54     or
55         when shutdownProcessor /\ wip = <>> /\ offset > Len(queue);
56         goto Done;
57     or
58         when \E id \in DOMAIN wip: (now - wip[id].start) >= Timeout;
59 +Cleanup:
60     with
61         id = CHOOSE id \in DOMAIN wip: (now - wip[id].start) >= Timeout,
62         off = wip[id].offset
63     do
64         processed := processed ++ off;
65         wip := Remove(wip, id);
66         committed := CommitOffset(processed, committed);
67         failed := failed ++ id;
68     end with;
69     or
```

CallHistory.tla v9->v10

```
47 CallComplete:
48     either
49         queue := Append(queue, [id |-> self, action |-> "complete"]);
50     or
51     -     failed := failed ++ self;
52     +     skip;
53     end either;

54     or
55         when shutdownProcessor /\ wip = <>> /\ offset > Len(queue);
56         goto Done;
57     or
58         when \E id \in DOMAIN wip: (now - wip[id].start) >= Timeout;
59 +Cleanup:
60     with
61         id = CHOOSE id \in DOMAIN wip: (now - wip[id].start) >= Timeout,
62         off = wip[id].offset
63     do
64         processed := processed ++ off;
65         wip := Remove(wip, id);
66         committed := CommitOffset(processed, committed);
67         failed := failed ++ id;
68     end with;
69     or
```

CallHistory.tla v9->v10

```
47 CallComplete:
48     either
49         queue := Append(queue, [id |-> self, action |-> "complete"]);
50     or
51     -     failed := failed ++ self;
52     +     skip;
53 end either;

88     or
89         when shutdownProcessor /\ wip = <>> /\ offset > Len(queue);
90         goto Done;
91     +     or
92     +     when \E id \in DOMAIN wip: (now - wip[id].start) >= Timeout;
93 +Cleanup:
94     +     with
95     +         id = CHOOSE id \in DOMAIN wip: (now - wip[id].start) >= Timeout,
96     +         off = wip[id].offset
97     +     do
98     +         processed := processed ++ off;
99     +         wip := Remove(wip, id);
100    +         committed := CommitOffset(processed, committed);
101    +         failed := failed ++ id;
102    +     end with;
103     or
```

CallHistory.tla v9->v10

```
1 | SPECIFICATION Spec
2 | /* Add statements after this line.
3 |
4 | CONSTANTS
5 |     Calls = {"call-1", "call-2", "call-3"}
6 |     Crashes = 1
7 |     Timeout = 10
8 |
9 | PROPERTIES
10|     Success
11|     Termination
```

CallHistory.tla v10 Run: Failed

```
23 Error: The following behavior constitutes a counter-example:  
531  
532 State 26: <ProcessCDR line 185, col 15 to line 200, col 79 of module CallHistory>  
533 /\ offset = 6  
534 /\ committed = 4  
535 /\ store = {"call-2", "call-3"}  
536 /\ now = 9  
537 /\ pc = ( "call-1" :> "Done" ;&  
538   "call-2" :> "Done" ;&  
539   "call-3" :> "Done" ;&  
540   "Shutdown" :> "Done" ;&  
541   "Processor" :> "Process" )  
542 /\ wip = ("call-1" :> [offset |-> 5, start |-> 9])  
543 /\ shutdownProcessor = TRUE  
544 /\ queue = << [id |-> "call-3", action |-> "setup"],  
545   [id |-> "call-2", action |-> "setup"],  
546   [id |-> "call-2", action |-> "complete"],  
547   [id |-> "call-3", action |-> "complete"],  
548   [id |-> "call-1", action |-> "setup"] >>  
549 /\ crashes = 1  
550 /\ failed = {}  
551 /\ processed = {1, 2, 3, 4}  
552  
553 State 27: Stuttering
```

CallHistory.tla v10 Run: Failed

```
23 | Error: The following behavior constitutes a counter-example:  
531 |  
532 | State 26: <ProcessCDR line 185, col 15 to line 200, col 79 of module CallHistory>  
533 | /\ offset = 6  
534 | /\ committed = 4  
535 | /\ store = {"call-2", "call-3"}  
536 | /\ now = 9  
537 | /\ pc = ( "call-1" :> "Done" ;&  
538 |   "call-2" :> "Done" ;&  
539 |   "call-3" :> "Done" ;&  
540 |   "Shutdown" :> "Done" ;&  
541 |   "Processor" :> "Process" )  
542 | /\ wip = ("call-1" :> [offset |-> 5, start |-> 9])  
543 | /\ shutdownProcessor = TRUE  
544 | /\ queue = << [id |-> "call-3", action |-> "setup"],  
545 |   [id |-> "call-2", action |-> "setup"],  
546 |   [id |-> "call-2", action |-> "complete"],  
547 |   [id |-> "call-3", action |-> "complete"],  
548 |   [id |-> "call-1", action |-> "setup"] >>  
549 | /\ crashes = 1  
550 | /\ failed = {}  
551 | /\ processed = {1, 2, 3, 4}  
552 |  
553 | State 27: Stuttering
```

CallHistory.tla v10 Run: Failed

```
23 | Error: The following behavior constitutes a counter-example:  
531 |  
532 | State 26: <ProcessCDR line 185, col 15 to line 200, col 79 of module CallHistory>  
533 | /\ offset = 6  
534 | /\ committed = 4  
535 | /\ store = {"call-2", "call-3"}  
536 | /\ now = 9  
537 | /\ pc = ( "call-1" :> "Done" /\  
538 |   "call-2" :> "Done" /\  
539 |   "call-3" :> "Done" /\  
540 |   "Shutdown" :> "Done" /\  
541 |   "Processor" :> "Process" )  
542 | /\ wip = ("call-1" :> [offset |-> 5, start |-> 9])  
543 | /\ shutdownProcessor = TRUE  
544 | /\ queue = << [id |-> "call-3", action |-> "setup"],  
545 |   [id |-> "call-2", action |-> "setup"],  
546 |   [id |-> "call-2", action |-> "complete"],  
547 |   [id |-> "call-3", action |-> "complete"],  
548 |   [id |-> "call-1", action |-> "setup"] >>  
549 | /\ crashes = 1  
550 | /\ failed = {}  
551 | /\ processed = {1, 2, 3, 4}  
552 |  
553 | State 27: Stuttering
```

CallHistory.tla v10->v11

```
86      or
87      when shutdownProcessor /\ wip = <>> /\ offset > Len(queue);
88      goto Done;
89 +     or
90 +     when /\ shutdownProcessor
91 +         /\ offset > Len(queue)
92 +         /\ wip /= <>>
93 +         /\ \A id \in DOMAIN wip: (now - wip[id].start) < Timeout;
94 +     skip;
95     or
```

CallHistory.tla v10->v11

```
86      or
87      when shutdownProcessor /\ wip = <>> /\ offset > Len(queue);
88      goto Done;
89 +     or
90 +     when /\ shutdownProcessor
91 +         /\ offset > Len(queue)
92 +             /\ wip /= <>>
93 +                 /\ \A id \in DOMAIN wip: (now - wip[id].start) < Timeout;
94 +     skip;
95     or
```

CallHistory.tla v10->v11

```
86      or
87      when shutdownProcessor /\ wip = <>> /\ offset > Len(queue);
88      goto Done;
89 +     or
90 +     when /\ shutdownProcessor
91 +         /\ offset > Len(queue)
92 +             /\ wip /= <>>
93 +                 /\ \A id \in DOMAIN wip: (now - wip[id].start) < Timeout;
94 +     skip;
95     or
```

CallHistory.tla v11 Run: Succeed

```
25 Model checking completed. No error has been found.  
26 Estimates of the probability that TLC did not check all reachable states  
27 because two distinct states had the same fingerprint:  
28 calculated (optimistic): val = 2.7E-10  
29 based on the actual fingerprints: val = 1.5E-8  
30 144036 states generated, 83106 distinct states found, 0 states left on queue.  
31 The depth of the complete state graph search is 49.
```

CallHistory.tla v11 Run: Succeed

```
25 Model checking completed. No error has been found.  
26 Estimates of the probability that TLC did not check all reachable states  
27 because two distinct states had the same fingerprint:  
28 calculated (optimistic): val = 2.7E-10  
29 based on the actual fingerprints: val = 1.5E-8  
30 144036 states generated, 83106 distinct states found, 0 states left on queue.  
31 The depth of the complete state graph search is 49.
```

CallHistory.tla v11 Run: Succeed

```
25 Model checking completed. No error has been found.  
26 Estimates of the probability that TLC did not check all reachable states  
27 because two distinct states had the same fingerprint:  
28 calculated (optimistic): val = 2.7E-10  
29 based on the actual fingerprints: val = 1.5E-8  
30 144036 states generated, 83106 distinct states found, 0 states left on queue.  
31 The depth of the complete state graph search is 49.
```

Модель:

ИТОГ

CallHistory.tla v11

```
1 ----- MODULE CallHistory -----
2
3 EXTENDS Sequences, Integers, TLC
4
5 CONSTANTS Calls
6 CONSTANTS Crashes
7 CONSTANTS Timeout
8
9 set ++ x == set \union {x}
10 Add(dict, key, val) == dict @\o (key :> val)
11 Remove(dict, key) == [i \in DOMAIN dict \ {key} |-> dict[i]]
12
13 (*--algorithm CallHistory
14
15 variables
16   queue = <<>>,
17   committed = 0,
18   shutdownProcessor = FALSE,
19   failed = {},
20   store = {};
21 define
22
23   \* {1, 3}, 0 -> 0
24   \* {2, 3}, 0 -> 0
```

CallHistory.tla v11

```
25  \* {1, 2, 3}, 0 -> 3
26  CommitOffset(set, current) ==
27      LET first == current + 1 IN
28      IF /\ first \in set
29          /\ \A i \in {j \in set: j > first}: i-1 \in set
30      THEN CHOOSE i \in set: i+1 \notin set
31      ELSE current
32
33  AllCallsProcessed == <>[](Calls \ failed = store)
34
35  Success == AllCallsProcessed
36
37 end define;
38
39 fair process Call \in Calls
40 begin CallSetup:
41     queue := Append(queue, [id |-> self, action |-> "setup"]);
42 CallComplete:
43     either
44         queue := Append(queue, [id |-> self, action |-> "complete"]);
45     or
46         skip;
47     end either;
48 end process;
```

CallHistory.tla v11

```
49
50 fair process Shutdown = "Shutdown"
51 begin SignalShutdown:
52     when \A c \in Calls: pc[c] = "Done";
53     shutdownProcessor := TRUE;
54 end process;
55
56 fair process Processor = "Processor"
57 variables
58     wip = <>>>,
59     processed = {},
60     crashes = 0,
61     now = 0,
62     offset = committed + 1;
63 begin Process:
64     while TRUE do
65         now := now + 1;
66         either
67             when Len(queue) >= offset;
68             ProcessCDR:
69                 with cdr = queue[offset]
70                 do
71                     if cdr.id \notin DOMAIN wip \wedge cdr.action = "setup" then
72                         wip := Add(wip, cdr.id, [start |-> now, offset |-> offset]);
```

CallHistory.tla v11

```
73      elsif cdr.id \in DOMAIN wip /\ cdr.action = "complete" then
74          processed := processed ++ wip[cdr.id].offset ++ offset;
75          wip := Remove(wip, cdr.id);
76          committed := CommitOffset(processed, committed);
77          store := store ++ cdr.id;
78      end if;
79  end with;
80  offset := offset + 1;
81 or
82 when shutdownProcessor /\ wip = <>> /\ offset > Len(queue);
83 goto Done;
84 or
85 when /\ shutdownProcessor
86     /\ offset > Len(queue)
87     /\ wip /= <>>
88     /\ \A id \in DOMAIN wip: (now - wip[id].start) < Timeout;
89 skip;
90 or
91 when \E id \in DOMAIN wip: (now - wip[id].start) >= Timeout;
92 Cleanup:
93 with
94     id = CHOOSE id \in DOMAIN wip: (now - wip[id].start) >= Timeout,
95     off = wip[id].offset
96 do
```

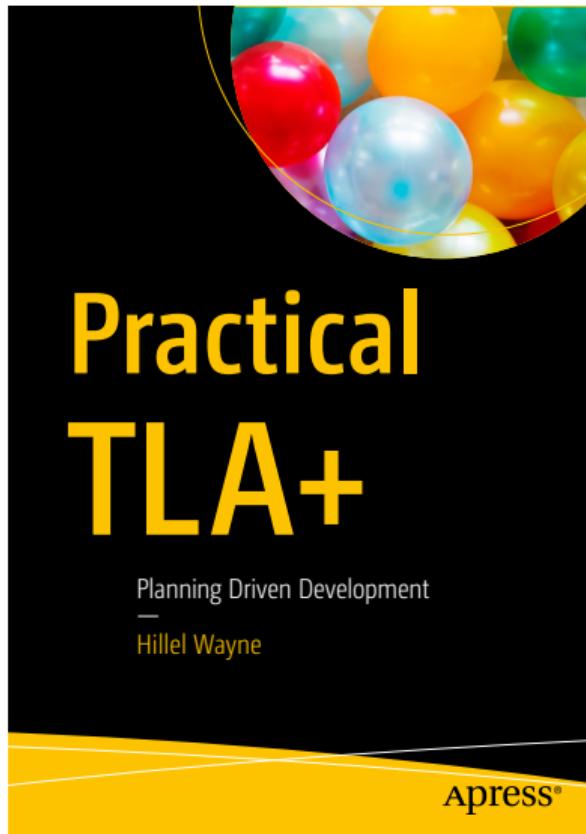
CallHistory.tla v11

```
97      processed := processed ++ off;
98      wip := Remove(wip, id);
99      committed := CommitOffset(processed, committed);
100     failed := failed ++ id;
101    end with;
102  or
103  when crashes < Crashes;
104 Crash:
105    crashes := crashes + 1 ||
106    processed := {} ||
107    offset := committed + 1 ||
108    wip := <<>>;
109   end either;
110  end while;
111 end process;
112
113 end algorithm; *)
```



©The Cartoon Network

С чего начать?



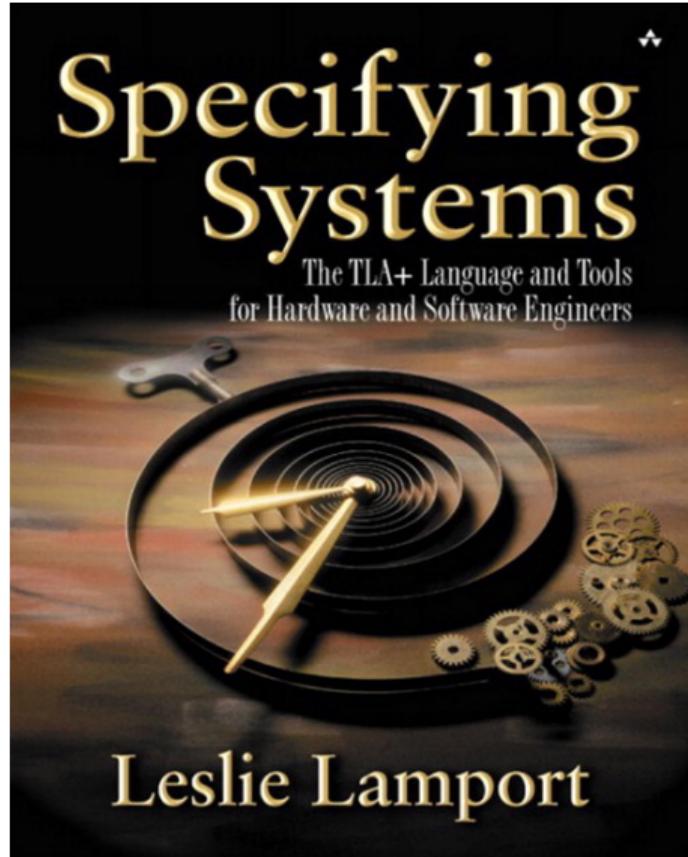
Practical TLA⁺

Planning Driven Development

by Hillel Wayne
Apress, 2018

Только про PlusCal,
идеально для вхождения в
тему!





Specifying Systems
The TLA⁺ Language and Tools
for Hardware and Software
Engineers
by Leslie Lamport
Addison-Wesley Professional, 2002

TLA⁺, обязательно иметь под
рукой

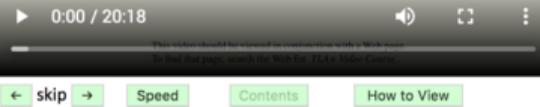


Introduction to TLA+

TLA+ Video Course – Lecture 1

Leslie Lamport

INTRODUCTION TO TLA+



Lecture 1

Web page: 20 March 2017

Video: 17 March 2017

Contents

- Prologue
- What is TLA+?
- Abstraction
- What Engineers Say
- What Can TLA+ Check?
- The Basic Abstraction
- State Machines
- A Tiny Example
- Epilogue

TLA⁺ Video Course

by Leslie Lamport

3.5 часа TLA⁺,

обязательно посмотреть!

(Лэмпорт прекрасный лектор!)



Lamport TLA⁺ Site: <https://lamport.azurewebsites.net/tla/tla.html>

Learn TLA⁺: <https://learntla.com>

google group: <https://groups.google.com/forum/#!forum/tlaplus>

reddit: <https://www.reddit.com/r/tlaplus/>

TLA⁺ Tools: <https://github.com/tlaplus/tlaplus>

Q&A

- TLA⁺ – это недели проектирования и месяцы кодирования?
 - Нет, это часы моделирования и дни или недели работы.
- Что насчет COQ (Agda, Idris, ...)?
 - Принципиально разная сложность
 - (есть еще Apalache (github.com/informalsystems/apalache) by Informal Systems)
- Что насчет Alloy?
 - Тоже отличный выбор для инженерных задач
- Можно ли из спецификации сразу делать код?
 - Можно, но не нужно (см. PGo (github.com/UBC-NSS/pgc))
 - (но если очень нужно, то есть Agda и Idris)

Спасибо!