Debugging Transactional Programs

Using the tm_db library

Yossi Lev

Brown University &
Sun Labs

Debugging Transactional Programs

Debugging Transactional Programs

- Have to do that

Debugging Transactional Programs

- Have to do that
- Can be easier than debugging general multithreaded programs
  - Isolation: think in pre and post conditions

Debugging Transactional Programs

- Have to do that
- Can be easier than debugging general multithreaded programs
  - Isolation: think in pre and post conditions

    ↳ Display *logical values*

Debugging Transactional Programs

- Have to do that
- Can be easier than debugging general multithreaded programs
  - Isolation: think in pre and post conditions

    ↳ Display *logical values*
  - No deadlocks

Debugging Transactional Programs

- Have to do that
- Can be easier than debugging general multithreaded programs
  - Isolation: think in pre and post conditions

    ↳ Display *logical values*
  - No deadlocks
  - Simpler programs, simpler debugging
- But, debuggers will have to change

Easier said than done...

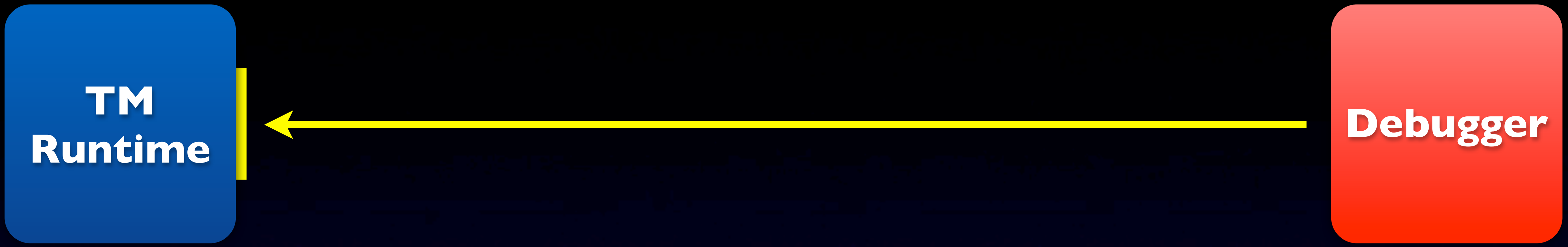
Easier said than done...

- Transactional Memory is still an open research area
  - Many different STM algorithms
    No clear agreement on "the right choice"
  - Hardware support in the future

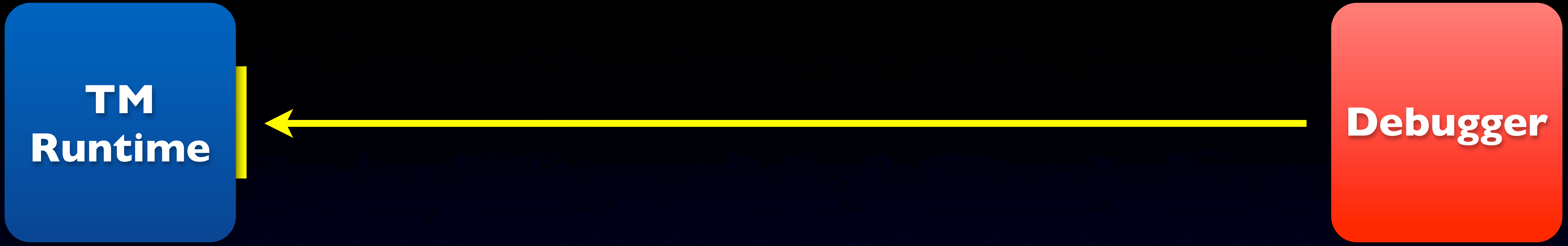
Easier said than done...

- Transactional Memory is still an open research area
  - Many different STM algorithms
    No clear agreement on "the right choice"
  - Hardware support in the future
  - Do not want to expose it to the user
  - Preferably not to the debugger either
- Abstraction layer: generic debugging infrastructure for transactional programs

Our Contribution

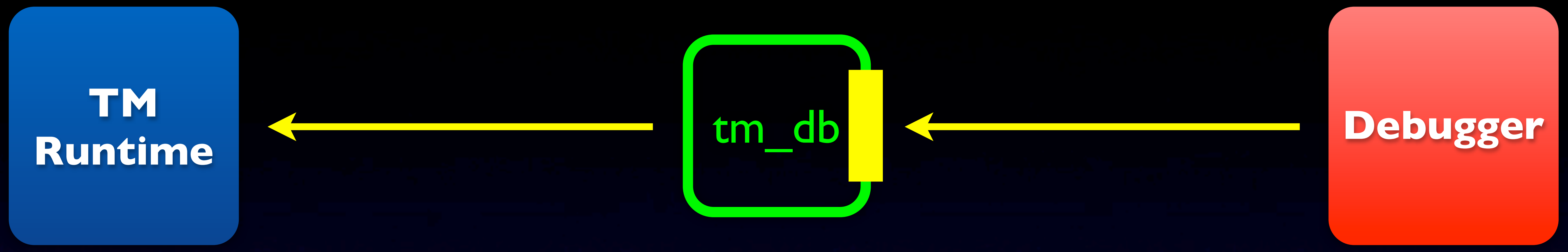


Our Contribution



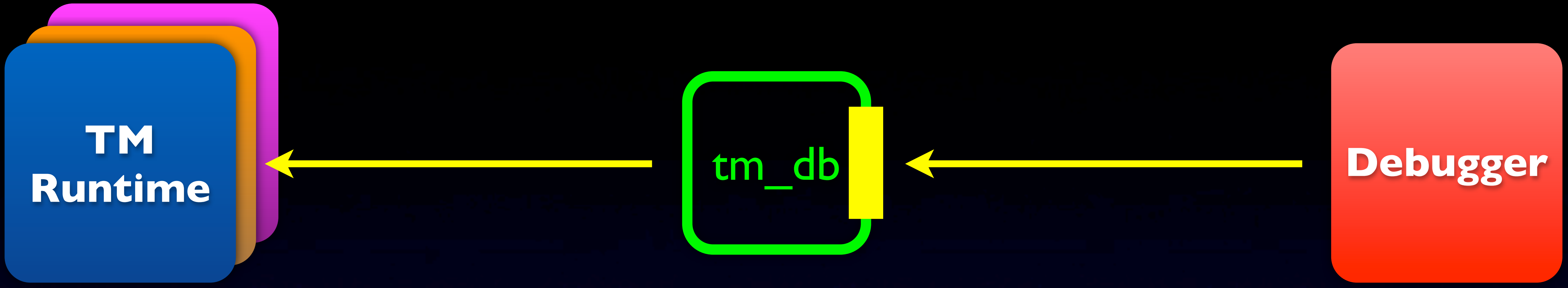
1. Interface for *generic transactional debugging support*
   - Abstract away details of TM runtime
   - Help TM designers keep debugging in mind
     - Gives TM runtimes a well-defined interface for transactional debugging

Our Contribution



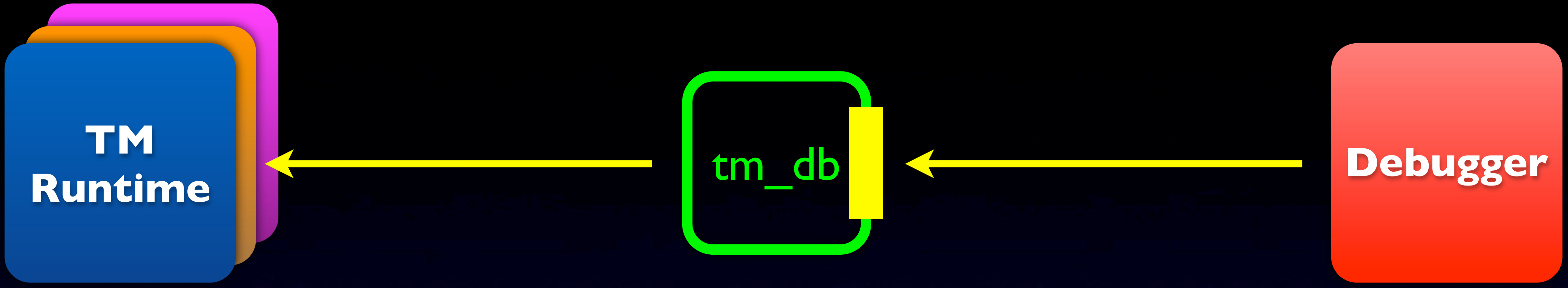
1. Interface for *generic transactional debugging support*
   - Abstract away details of TM runtime
   - Help TM designers keep debugging in mind
     - Gives TM runtimes a well-defined interface for transactional debugging
2. The tm_db library: implements this interface

Our Contribution



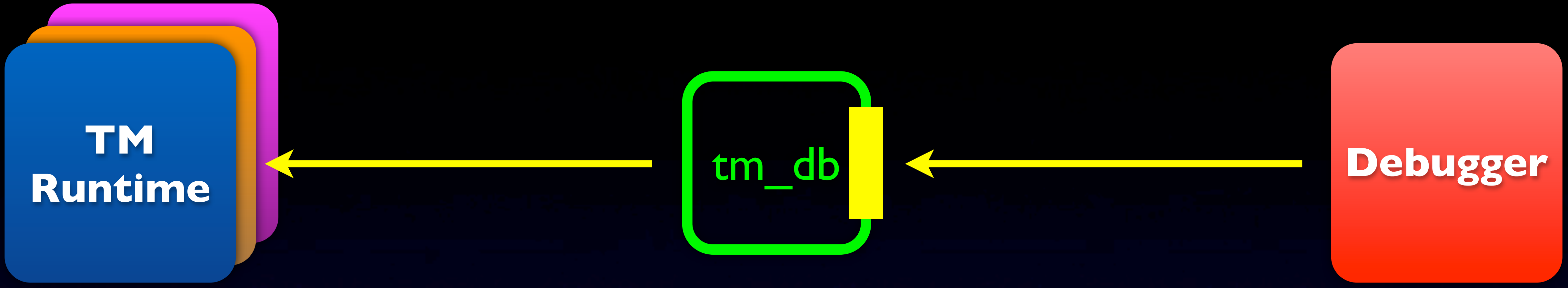
1. Interface for *generic transactional debugging support*
    - Abstract away details of TM runtime
    - Help TM designers keep debugging in mind
        - Gives TM runtimes a well-defined interface for transactional debugging
2. The tm_db library: implements this interface
    - Can support multiple TM runtimes

Our Contribution



1. Interface for *generic transactional debugging support*
   - Abstract away details of TM runtime
   - Help TM designers keep debugging in mind
     - Gives TM runtimes a well-defined interface for transactional debugging
2. The tm_db library: implements this interface
   - Can support multiple TM runtimes
   - Supports multiple debuggers
   - Open source: experiment with new debugging features

Agenda



- "Generic transactional debugging support"
  - What features?
  - In what sense are they generic?
- Library design and implementation

Examining Memory

Examining Memory

- Would like to show values from the point of view of the debugged program

Examining Memory

- Would like to show values from the point of view of the debugged program
  - Logical values: expose the whole transaction’s effect atomically at the time it takes effect

Examining Memory

- Would like to show values from the point of view of the debugged program
  - Logical values: expose the whole transaction’s effect atomically at the time it takes effect
  - Examine write set to show tentative values of currently debugged transaction

Transactions IDs

- Three notions of interest
  - Atomic block
    lexical scope
  - Logical transaction
    successful execution of an atomic block
  - Physical transaction
    system level: an attempt to execute a logical transaction

Transactions IDs

- Three notions of interest
  - Atomic block
    lexical scope
  - Logical transaction
    successful execution of an atomic block
  - Physical transaction
    system level: an attempt to execute a logical transaction

```
Tx Id: <31, 4159, 3>
```

$\uparrow$

Thread Id

Transactions IDs

- Three notions of interest
  - Atomic block
    lexical scope
  - Logical transaction
    successful execution of an atomic block
  - Physical transaction
    system level: an attempt to execute a logical transaction

```
Tx Id: <31, 4159, 3>
```

Thread Id

Transactions IDs

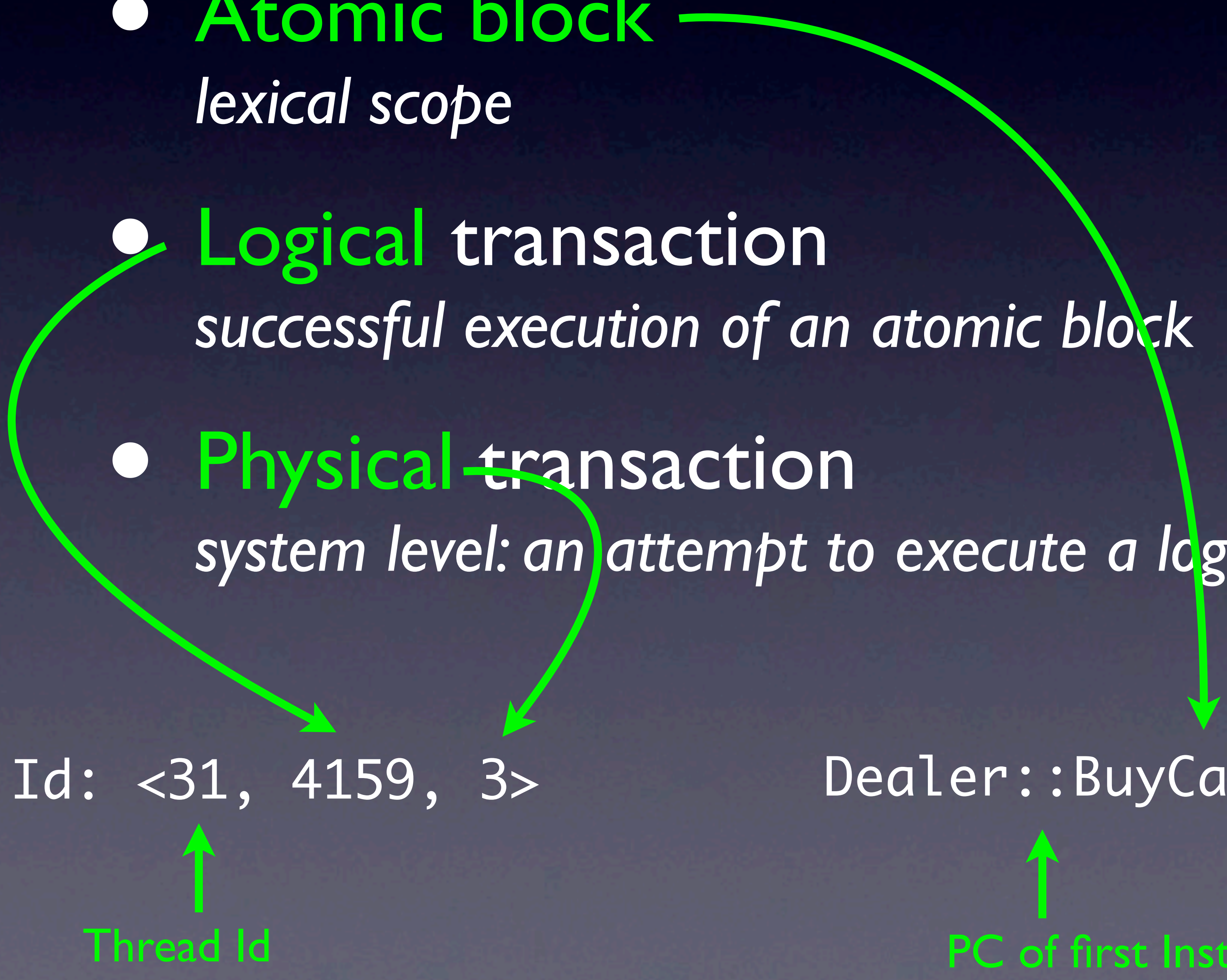
- Three notions of interest
  - Atomic block
    lexical scope
  - Logical transaction
    successful execution of an atomic block
  - Physical transaction
    system level: an attempt to execute a logical transaction

```
Tx Id: <31, 4159, 3>
```

Thread Id

Transactions IDs

- Three notions of interest
  - Atomic block
    lexical scope
  - Logical transaction
    successful execution of an atomic block
  - Physical transaction
    system level: an attempt to execute a logical transaction



```
Tx Id: <31, 4159, 3>          Dealer::BuyCar+0x10
```

Transactions IDs

- Three notions of interest
  - Atomic block
    lexical scope
  - Logical transaction
    successful execution of an atomic block
  - Physical transaction
    system level: an attempt to execute a logical transaction

```
Tx Id: <31, 4159, 3>          Dealer::BuyCar+0x10
Tx Id: <31, 4159, 5>          Dealer::BuyCar+0x10
```

Transaction Status

- Tx Status
  Active, Invalid, Aborted, Committed

Transaction Status

- Tx Status
  Active, Invalid, Aborted, Committed

Transaction Status

- Tx Status
  Active, $\boxed{\text{Invalid}}$, Aborted, Committed

Transaction Status

- Tx Status
  Active, Invalid, Aborted, Committed

Transaction Status

- Tx Status
  Active, Invalid, Aborted, $\boxed{\text{Committed}}$

Transaction Status

- Tx Status
  Active, Invalid, Aborted, Committed

Tx Id	Atomic Block	Status
<2,1907,0>	PushHead+0x8	Committed
<3,2217,1>	PushTail+0x8	Invalid
<4,2082,0>	PushHead+0x8	Aborted
<5,2107,0>	PushHead+0x8	Active
<6,2210,0>	PushTail+0x8	Invalid

Completed Transaction

Completed Transaction

- Report information of latest transaction
  even if already completed

Completed Transaction

- Report information of latest transaction
  even if already completed
- Status query also reports whether completed

Tx Id	Atomic Block	Status	
<2,1907,0>	PushHead+0x8	Committed	(Completed)
<3,2217,2>	PushTail+0x8	Committed	
<4,2082,0>	PushHead+0x8	Aborted	(Completed)
<5,2107,0>	PushHead+0x8	Active	
<6,2210,0>	PushTail+0x8	Invalid	

Access and Coverage

- Read and Write Accesses:
  examine set of read and written locations

Access and Coverage

- Read and Write Accesses:
  examine set of read and written locations
- Not enough due to false conflicts
  - Tx accesses location L
    conflicts with accessed to more than just location L
  - Capture by Coverage

Access and Coverage

- Library provides:
  - Iterator over accessed locations
  - Query coverage
- Reported at user access time

Access and Coverage

- Library provides:
  - Iterator over accessed locations
  - Query coverage
- Reported at user access time
- Find conflicts on a variable

```
> tx coverage &g_list->next->key

Tx Id        Atomic Block     Status       Coverage   Read   Written
<2,1907,0>   Insert+0x8       Active       R & W       ✓       ✓
<3,2217,1>   Insert+0x8       Active       R & W       ✓
<4,2082,0>   Insert+0x8       Active       R & W
```

Access and Coverage

- Library provides:
    - Iterator over accessed locations
    - Query coverage
- Reported at user access time
- Show *all* conflicts

```
> tx conflicts

Address     Tx Id        Atomic Block  Status   Coverage Read Written
0x10019dc48 <2,1907,0>  Insert+0x8    Active   R & W      ✓      ✓
            <3,2217,1>  Insert+0x8    Active   R & W      ✓
            <4,2082,0>  Insert+0x8    Active   R & W
0x10018bde0 <2,1907,0>  Insert+0x8    Active   R          ✓
            <3,2217,1>  Insert+0x8    Active   W                 ✓
```

Transactional Events

- Track/Stop at transaction related events: TxBegin, TxCommit, TxAbort, TxAbortOther

Transactional Events

- Track/Stop at transaction related events: TxBegin, TxCommit, TxAbort, TxAbortOther
- Run time filtering
  avoid stopping the program unnecessarily

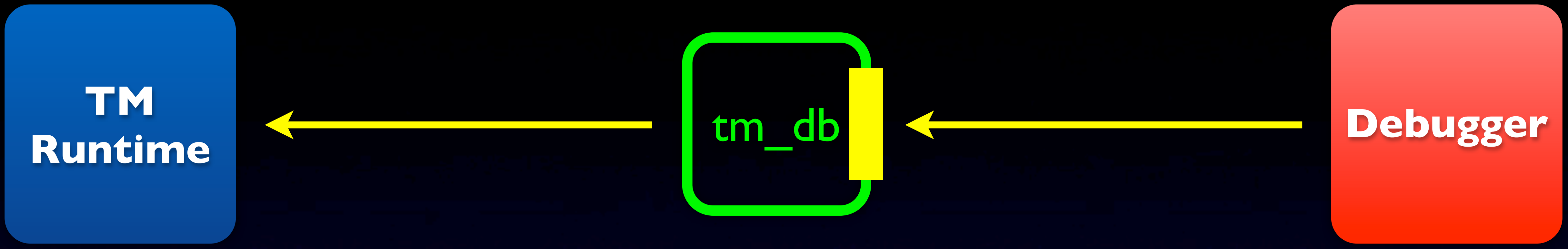
Transactional Events

- Track/Stop at transaction related events: TxBegin, TxCommit, TxAbort, TxAbortOther
- Run time filtering
  avoid stopping the program unnecessarily
  - Per thread monitoring
    stop if the event occurs for a given thread
  - Other monitoring parameters
    stop if
    - aborted for a specific reason (e.g. self-abort)
    - aborted-other due to specific conflict type (e.g. RW)

Transactional Events

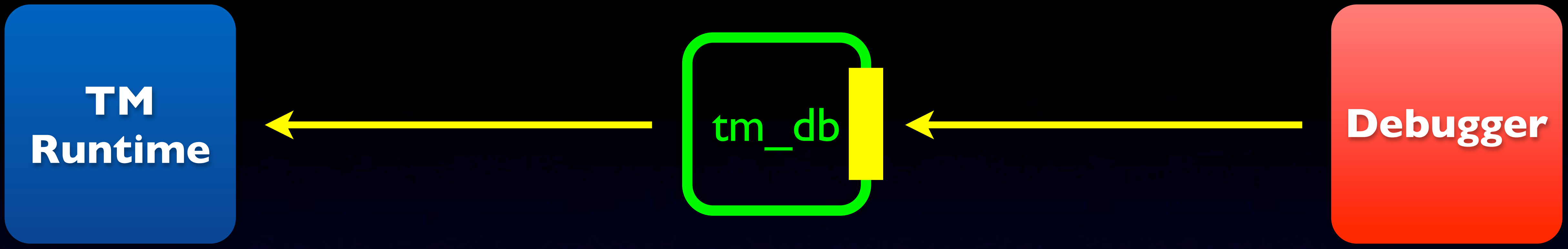
- Track/Stop at transaction related events: TxBegin, TxCommit, TxAbort, TxAbortOther
- Run time filtering
  avoid stopping the program unnecessarily
  - Per thread monitoring
    stop if the event occurs for a given thread
  - Other monitoring parameters
    stop if
    - aborted for a specific reason (e.g. self-abort)
    - aborted-other due to specific conflict type (e.g. RW)
  - Monitoring Scopes

Agenda



- "Generic transactional debugging support"
  - What features?
  - In what sense are they generic?

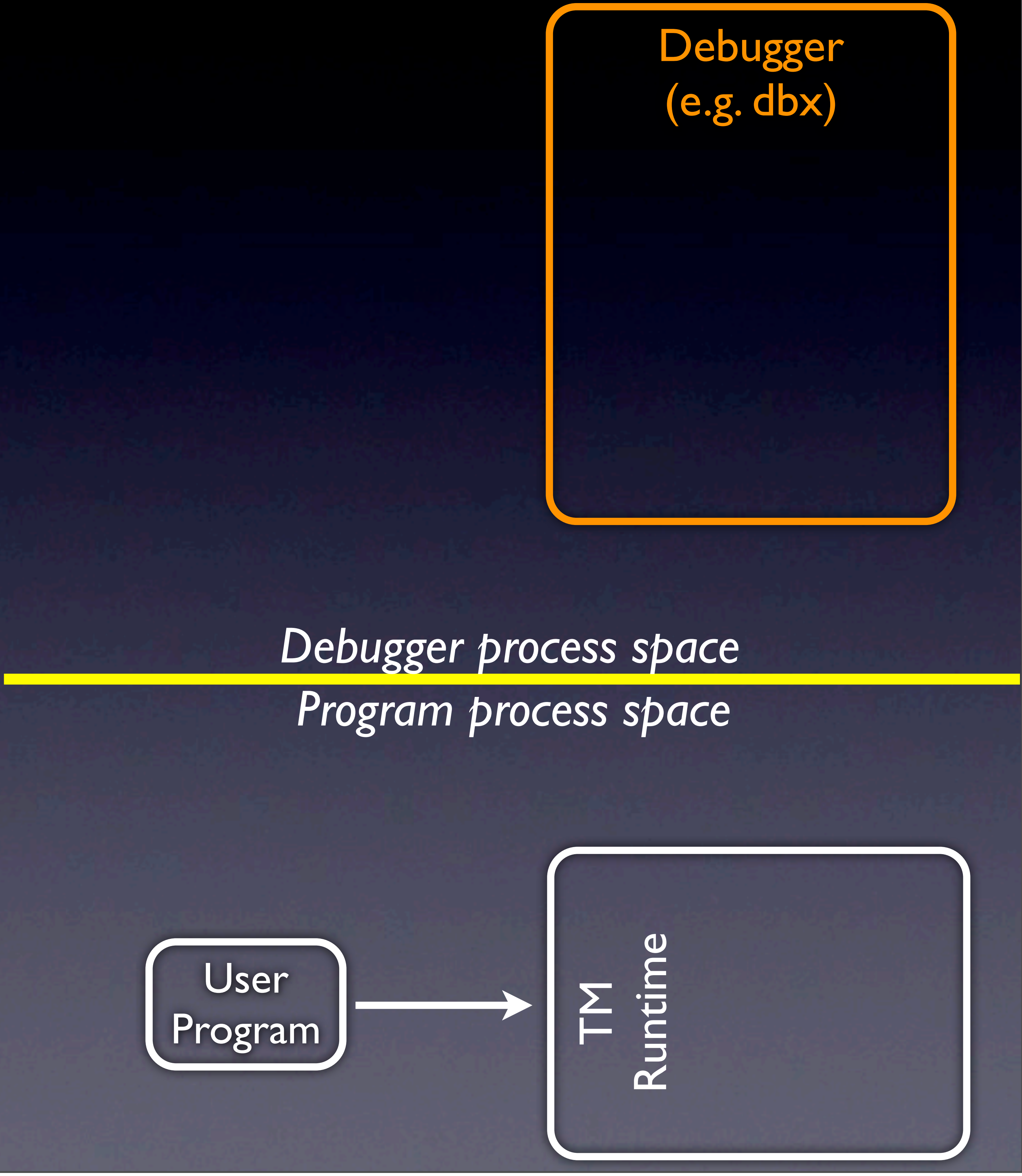
Agenda



- "Generic transactional debugging support"
  - What features?
  - In what sense are they generic?
- Library design and implementation

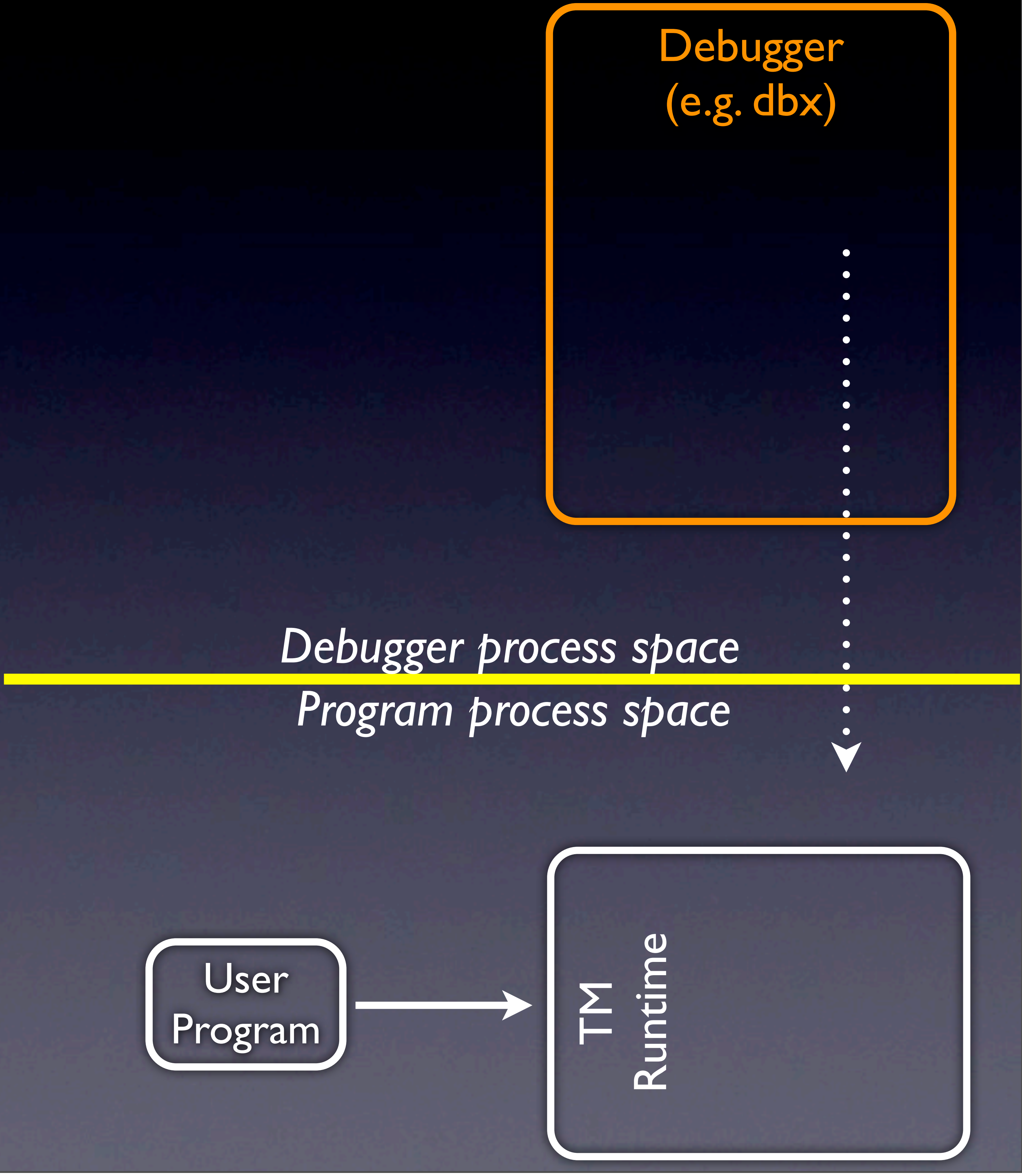
tm_db: Solution Design

- Program and debugger ran by different processes
- proc_service: provides external libraries access to the debugged process
  - mostly read/write remote memory
- RDM:
  Remote Debugging Module
  TM-specific part of the solution



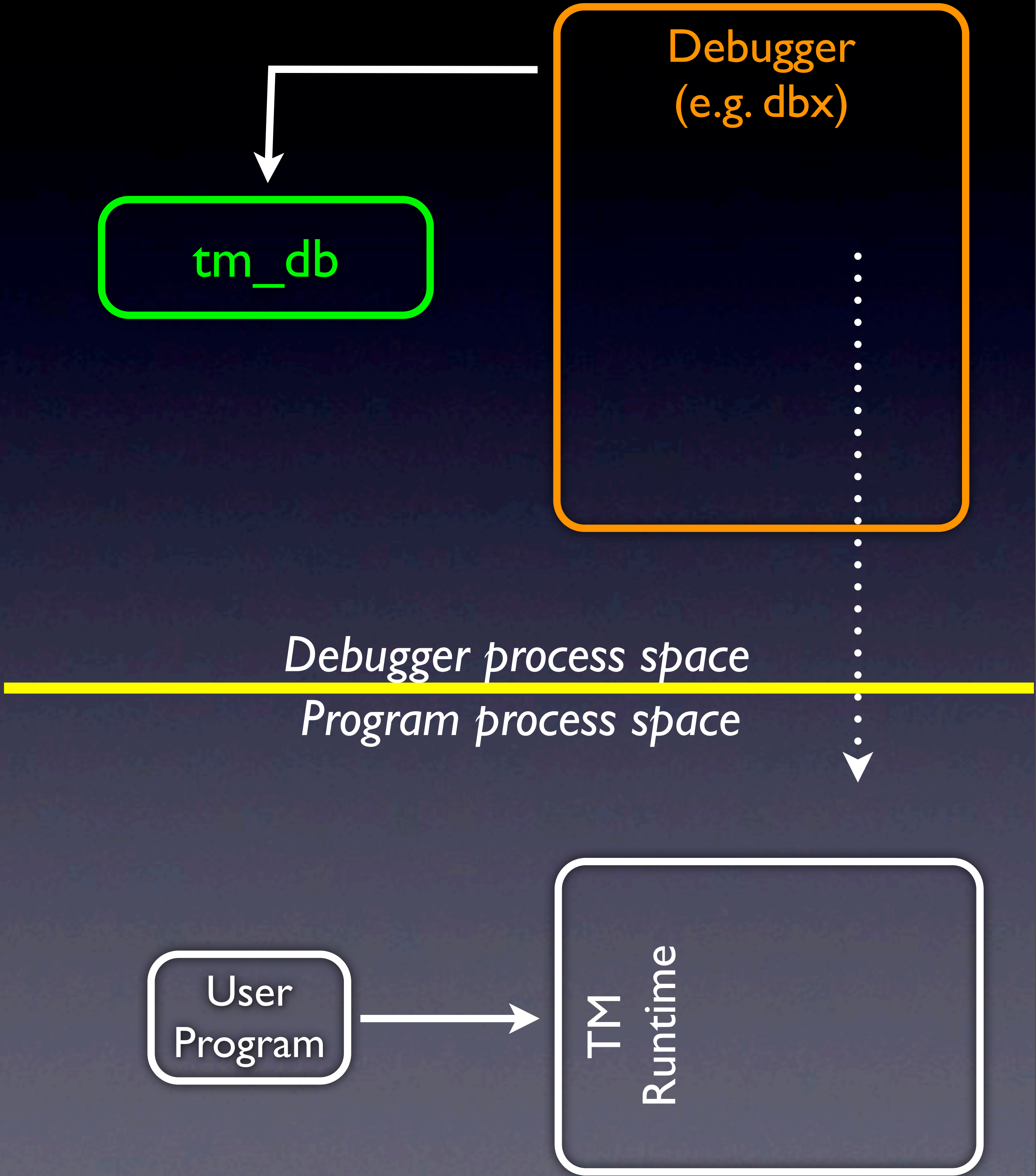
tm_db: Solution Design

- Program and debugger ran by different processes
- proc_service: provides external libraries access to the debugged process
  - mostly read/write remote memory
- RDM:
  Remote Debugging Module
  TM-specific part of the solution



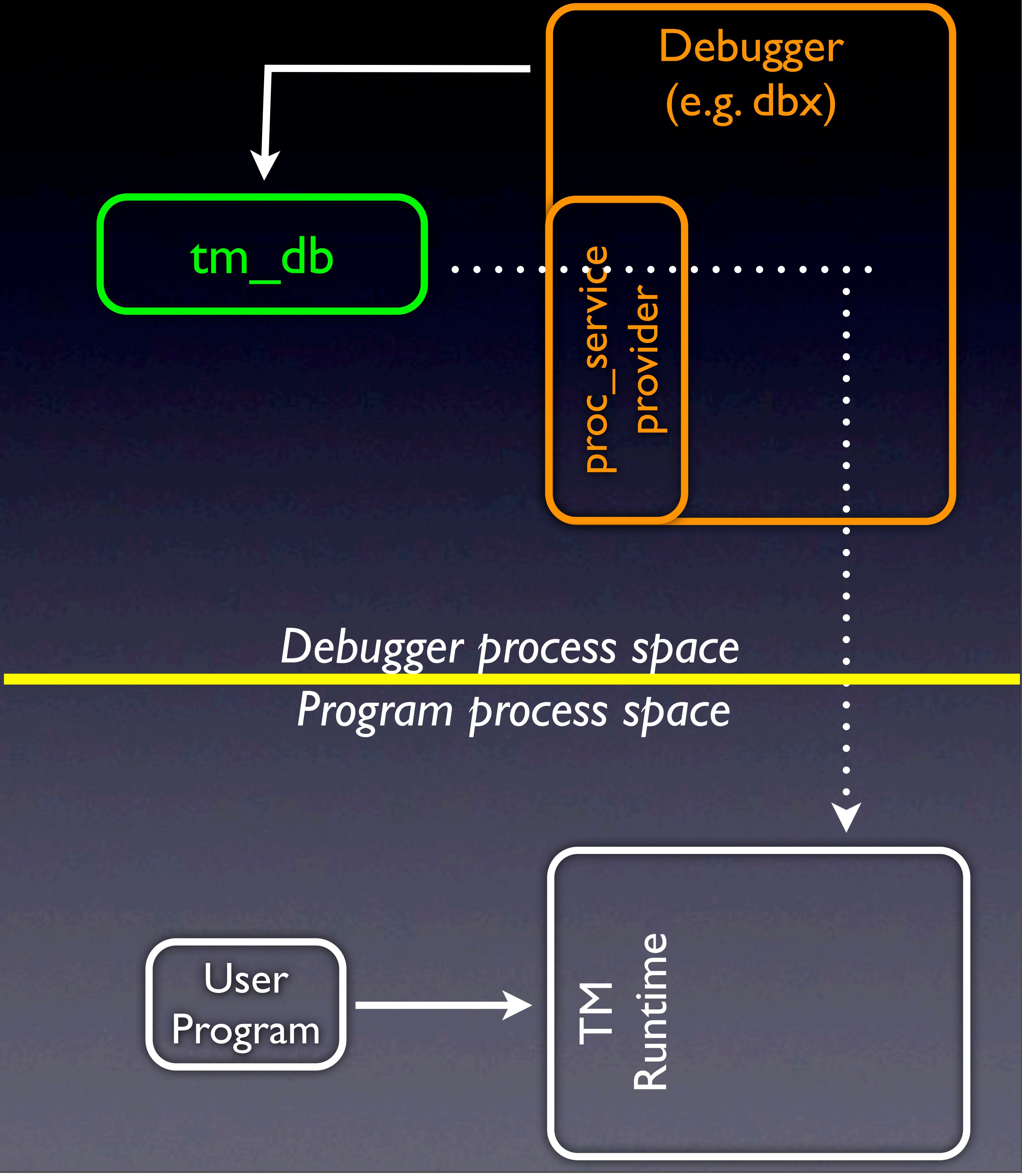
tm_db: Solution Design

- Program and debugger ran by different processes
- proc_service: provides external libraries access to the debugged process
    - mostly read/write remote memory
- RDM:
  Remote Debugging Module
  TM-specific part of the solution



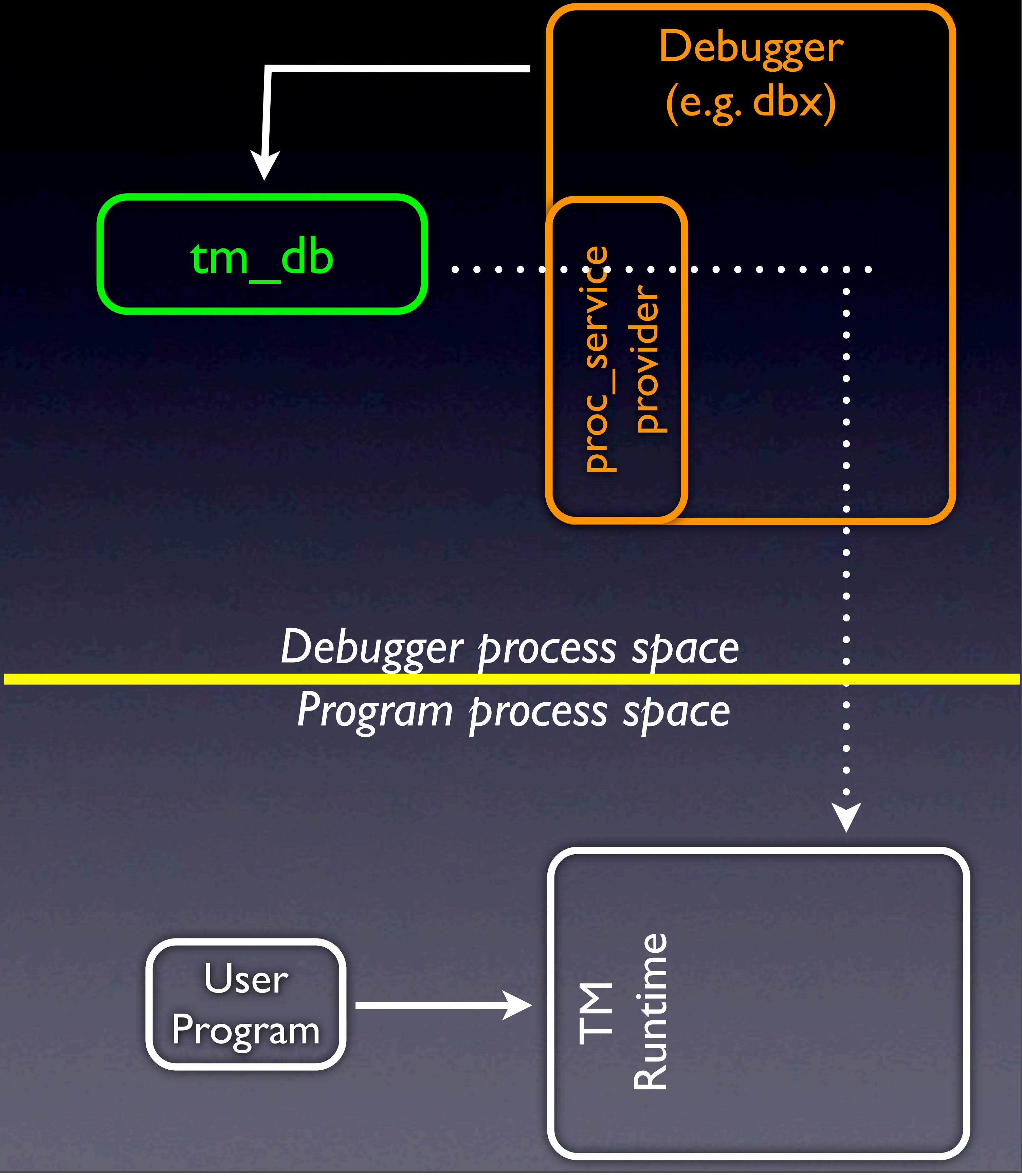
tm_db: Solution Design

- Program and debugger ran by different processes
- proc_service: provides external libraries access to the debugged process
    - mostly read/write remote memory



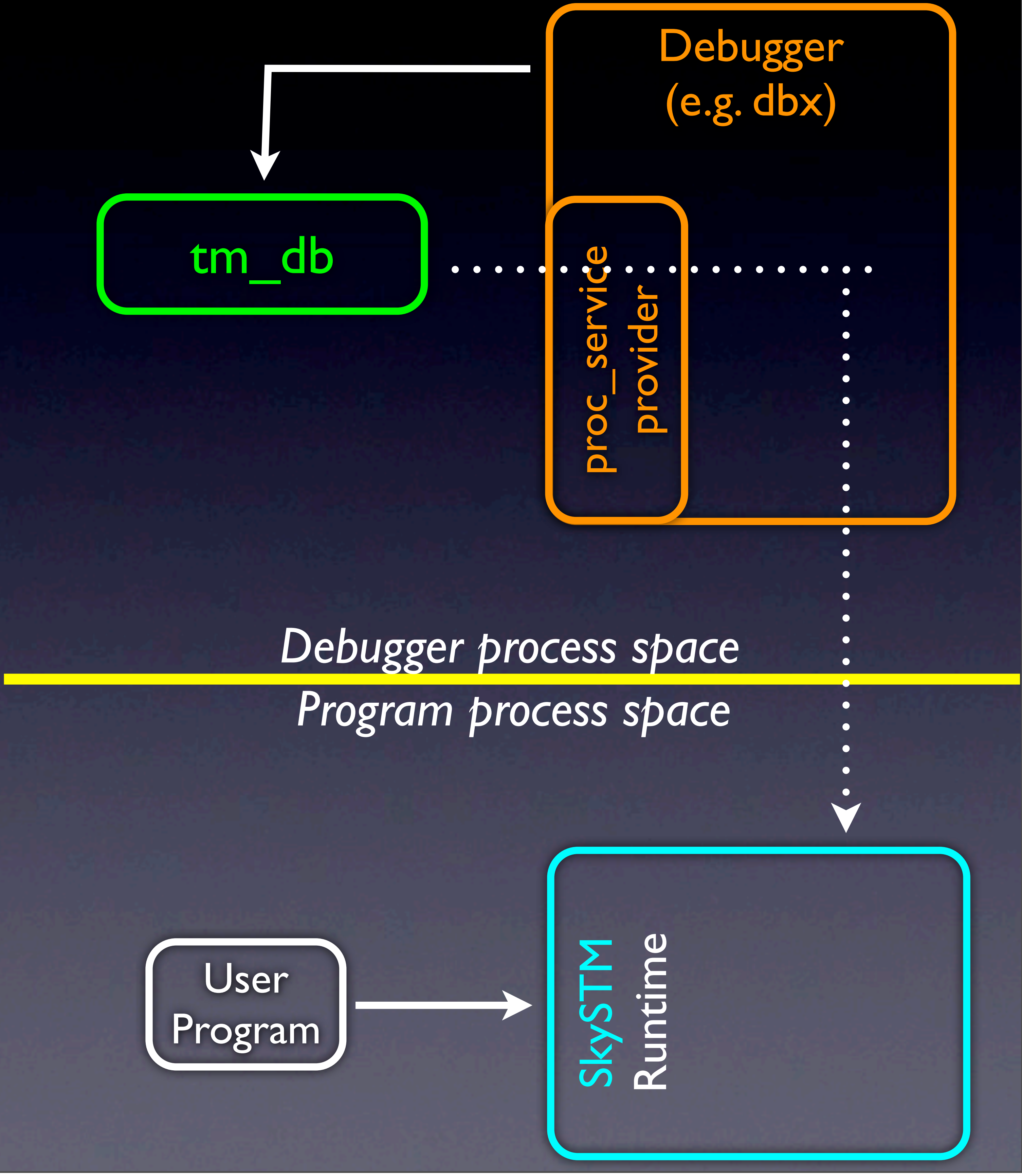
tm_db: Solution Design

- Program and debugger ran by different processes
- proc_service: provides external libraries access to the debugged process
  - mostly read/write remote memory
- RDM:
  Remote Debugging Module
  TM-specific part of the solution



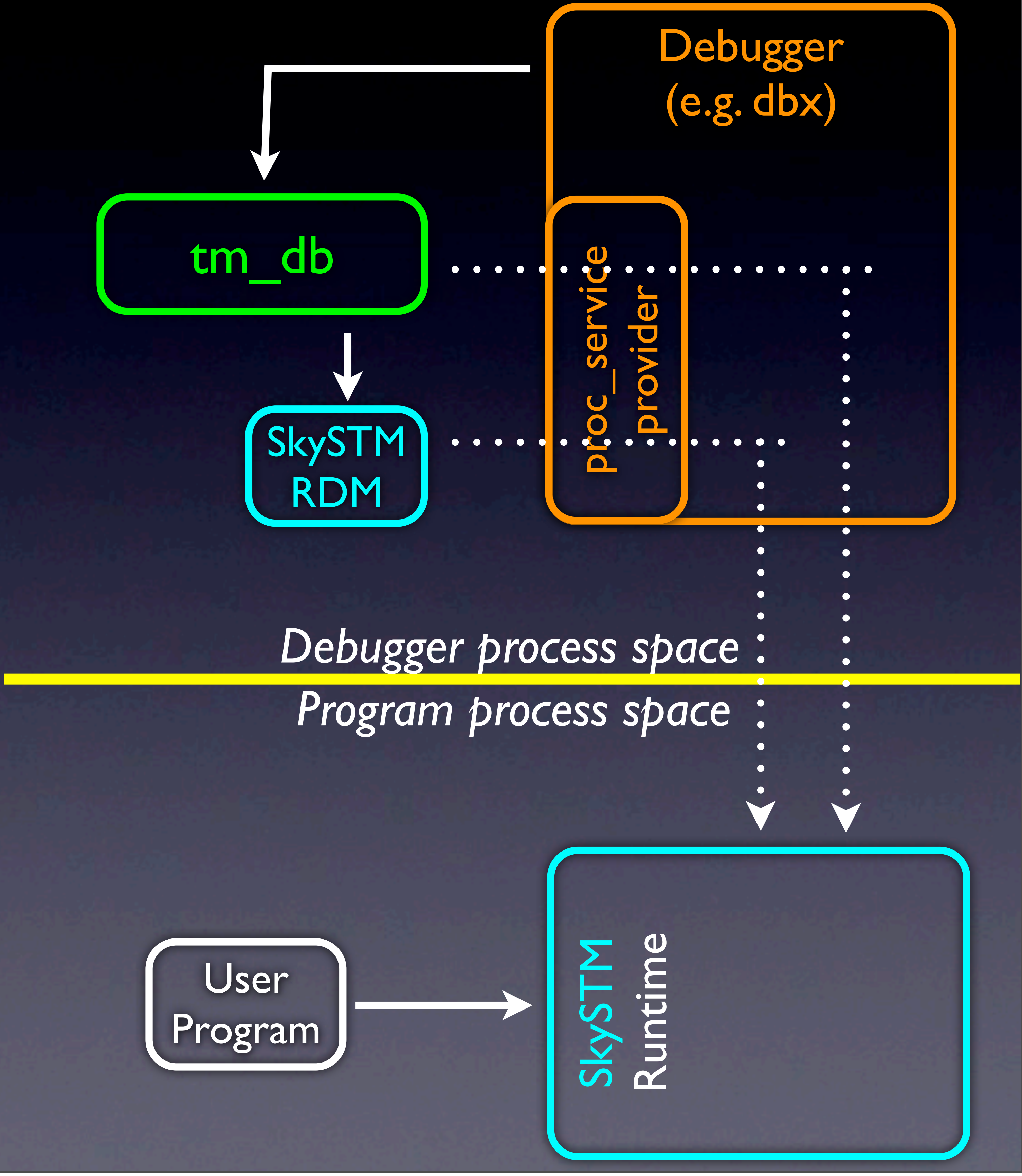
tm_db: Solution Design

- Program and debugger ran by different processes
- proc_service: provides external libraries access to the debugged process
  - mostly read/write remote memory
- RDM:
  Remote Debugging Module
  TM-specific part of the solution



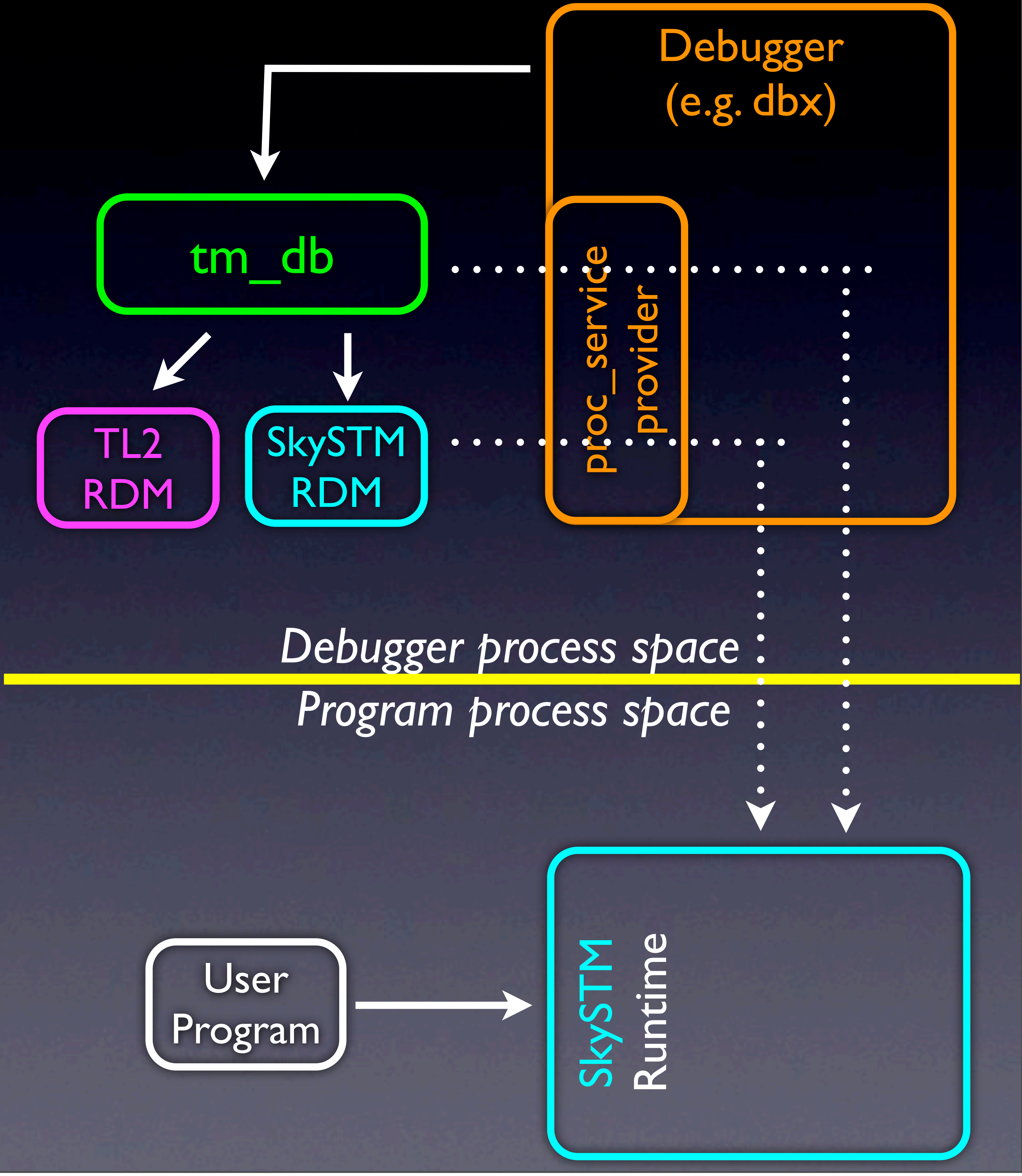
tm_db: Solution Design

- Program and debugger ran by different processes
- proc_service: provides external libraries access to the debugged process
  - mostly read/write remote memory
- RDM:
  Remote Debugging Module
  TM-specific part of the solution



tm_db: Solution Design

- Program and debugger ran by different processes
- proc_service: provides external libraries access to the debugged process
  - mostly read/write remote memory
- RDM:
  Remote Debugging Module
  TM-specific part of the solution



Notes

Notes

- Partial functionality is Ok
  - Not all features must be implemented by all runtimes

Notes

- Partial functionality is Ok
    - Not all features must be implemented by all runtimes
- Future work
    - Performance Debugging

Notes

- Partial functionality is Ok
    - Not all features must be implemented by all runtimes
- Future work
    - Performance Debugging
    - Graduating

Summary

Summary

- Debugging transactional programs
  should not be difficult
  - But debuggers has to change

Summary

- Debugging transactional programs should not be difficult
  - But debuggers has to change
- We define an interface for generic transactional debugging support
- Provide a library that implements it
- Opens the door for transactional debugging support with various debuggers and runtimes