



Oracle OpenWorld
September 30 - October 4, 2012
San Francisco




Resolving Child Cursor Issues Resulting In Mutex Waits

Martin Klier
Senior DBA
Klug GmbH integrierte Systeme

San Francisco, Sept 30th, 2012



Agenda

- Introduction 
- Oracle Parsing, Child Cursors
- Mutexes, Waits and Reasons
- Issues, Quick Fixes and Solutions
- Summary, Acknowledgements, Q&A session



Oracle OpenWorld
September 30 - October 4, 2012
San Francisco

Speaker

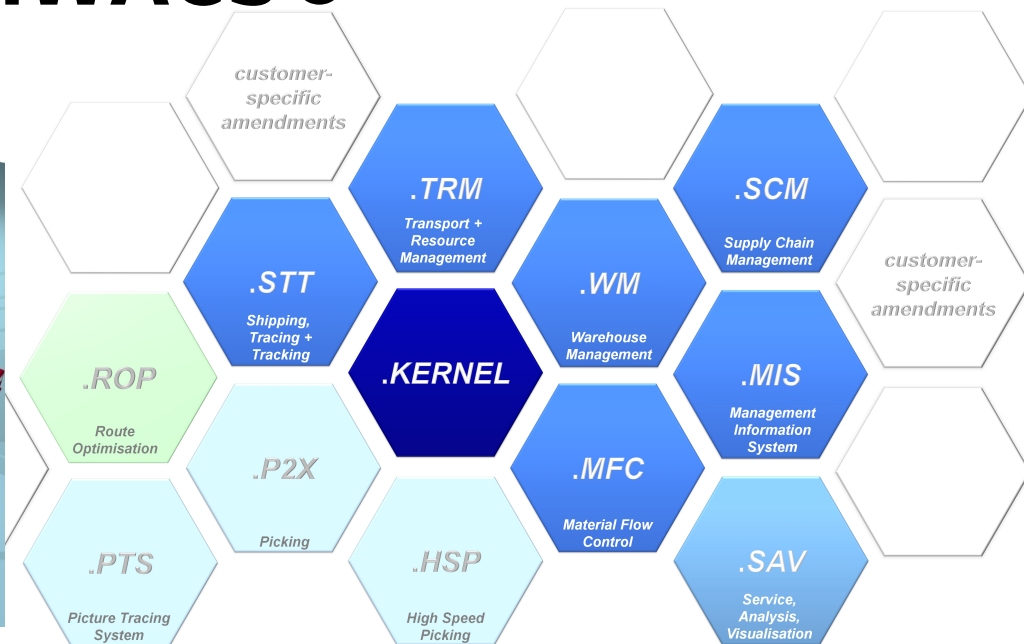
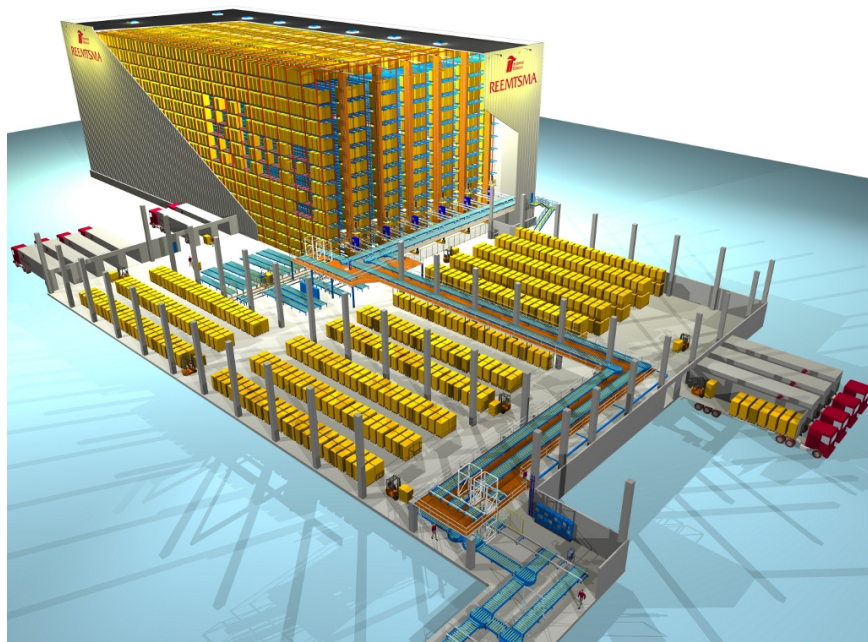


- Martin Klier (martin.klier@klug-is.de)
- Senior DBA for **ORACLE**® at  **Klug**
integrierte Systeme
- Focus on Performance, Tuning and High Availability
- Linux since 1997, Oracle since 2003
- Weblog: <http://www.usn-it.de>


- Klug GmbH integrierte Systeme (<http://www.klug-is.de>)
92552 Teunz, GERMANY
- Specialist leading in complex intralogistical solutions
- Planning and design of automated intralogistics systems,
focus on software and system control / PLC
- >300 successful major projects in Europe, America, Asia



iWACS®

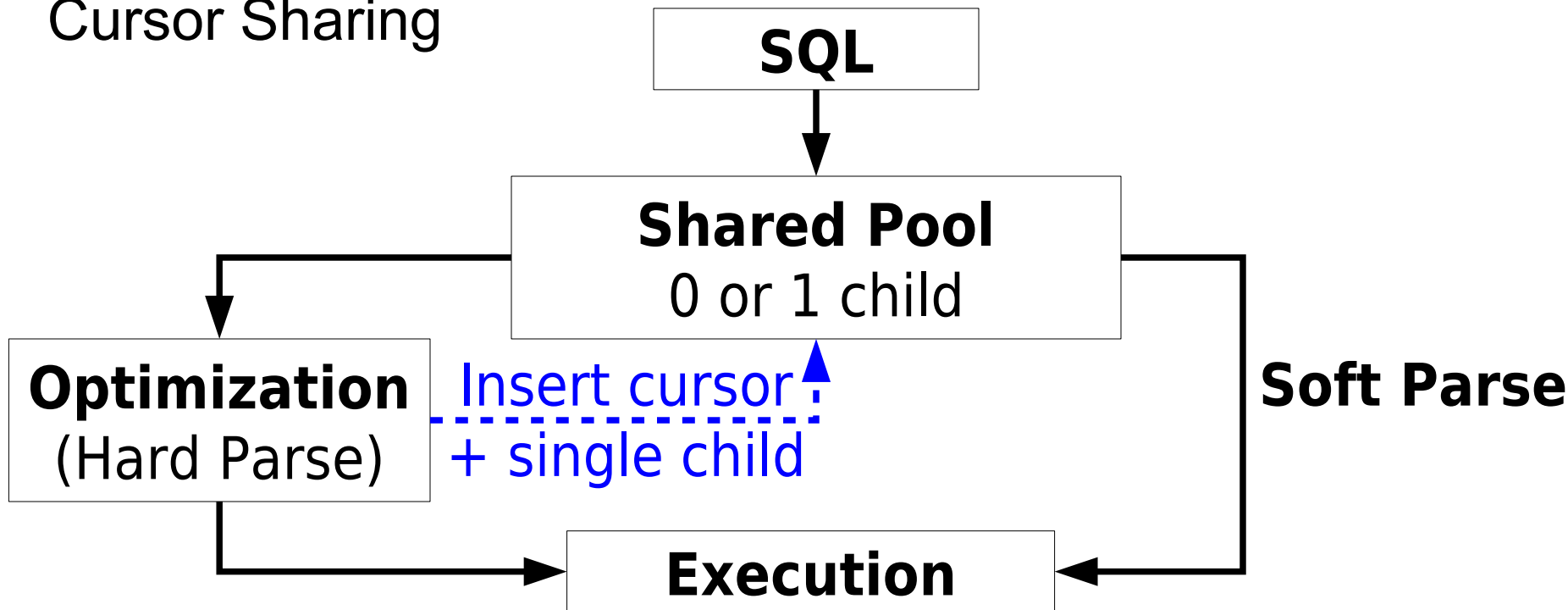


Agenda

- Introduction
- Oracle Parsing, Child Cursors 
- Mutexes, Waits and Reasons
- Issues, Quick Fixes and Solutions
- Summary, Acknowledgements, Q&A session

Parsing

Cursor Sharing



Parsing

Adaptive
Cursor Sharing I

SQL

Bind Mismatch
or ~62
other Reasons

Shared Pool
1 or n children

**Cardinality
Feedback**

Soft Parse

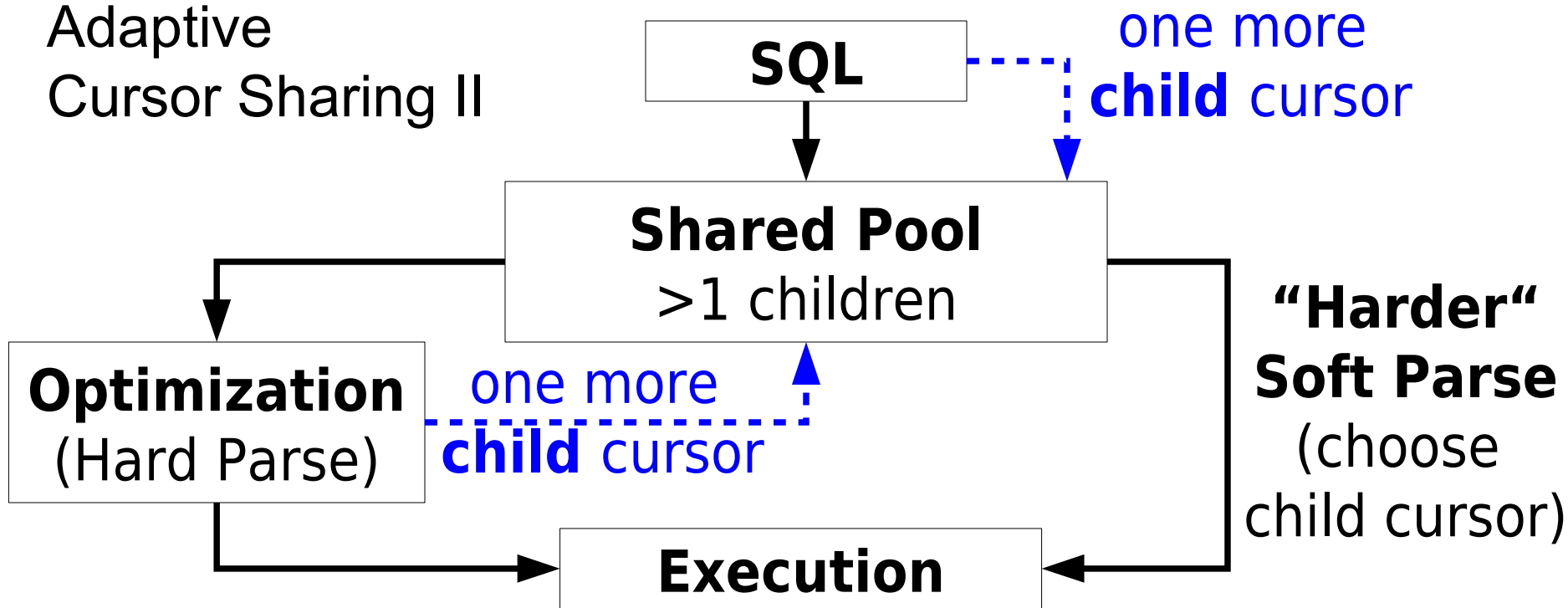
Execution

64 Reasons for not re-using an existing child cursor

- Optimizer mode change (ALL_ROWS, FIRST_ROWS)
- NLS- and user identity trouble,
- Outline mismatch, Cardinality feedback (**wanted**)
- Bind mismatch (many sub-reasons, **most unwanted**)
- ...

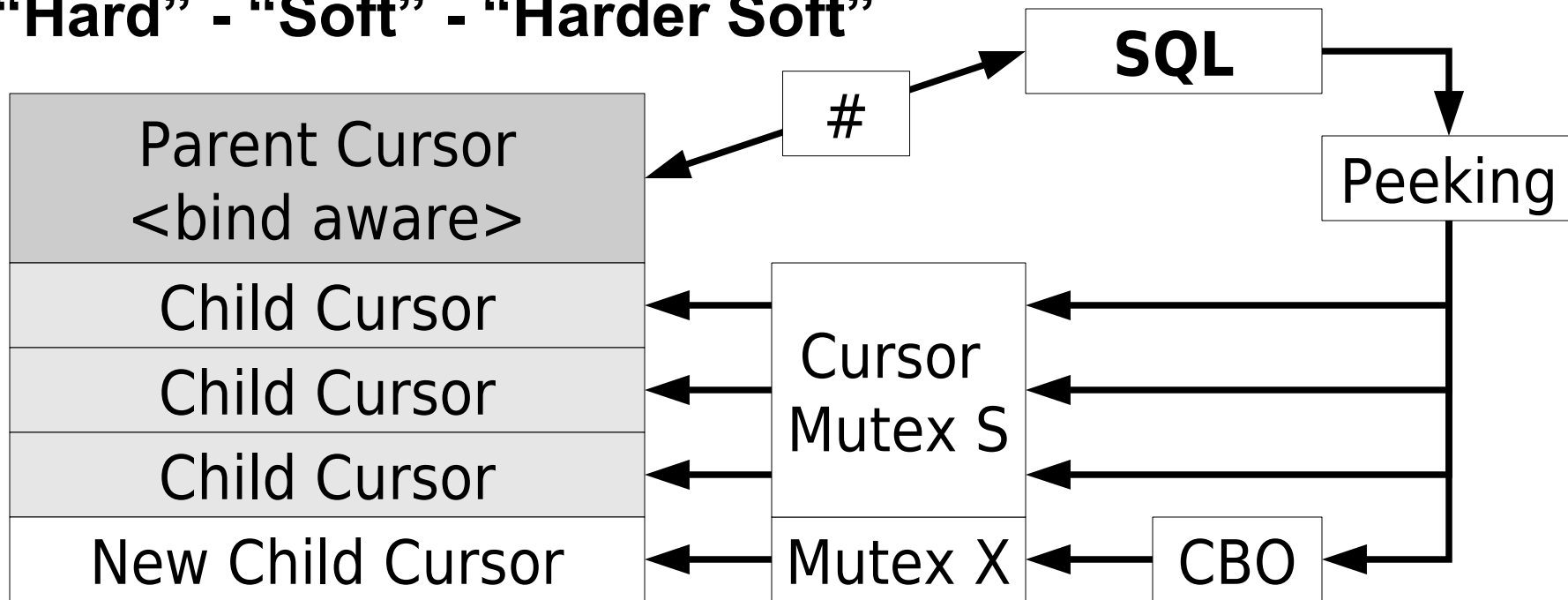
Parsing

Adaptive
Cursor Sharing II




Parsing

“Hard” - “Soft” - “Harder Soft”

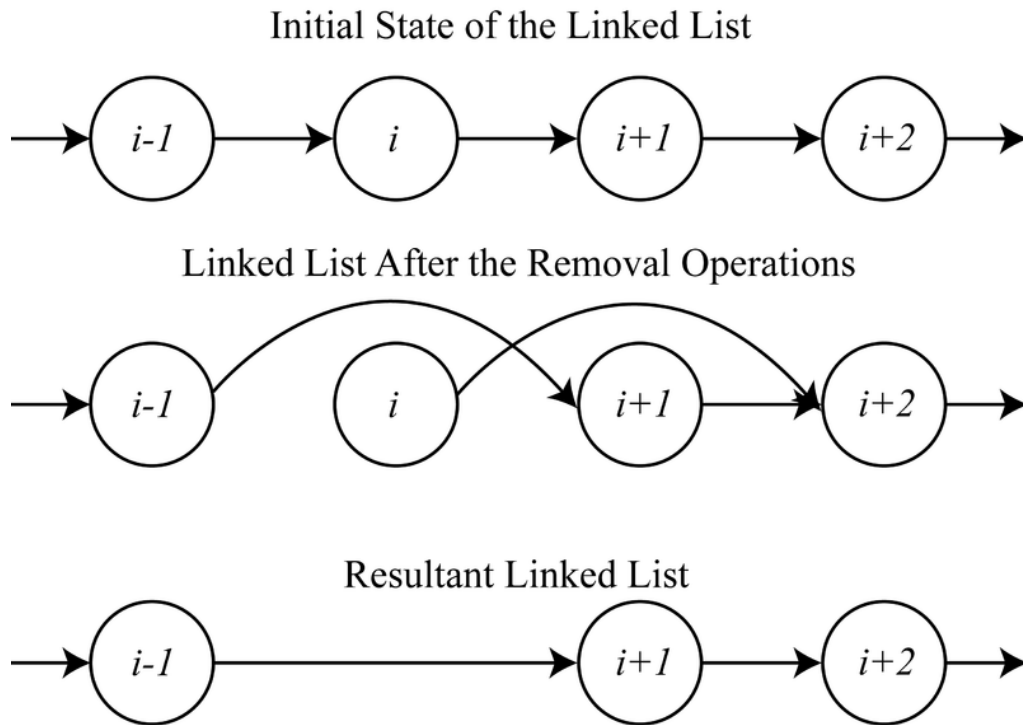


Agenda

- Introduction
- Oracle Parsing, Child Cursors
- Mutexes, Waits and Reasons 
- Issues, Quick Fixes and Solutions
- Summary, Acknowledgements, Q&A session

Mutex

Example:
Simultaneously
removing two nodes
from a singly linked list



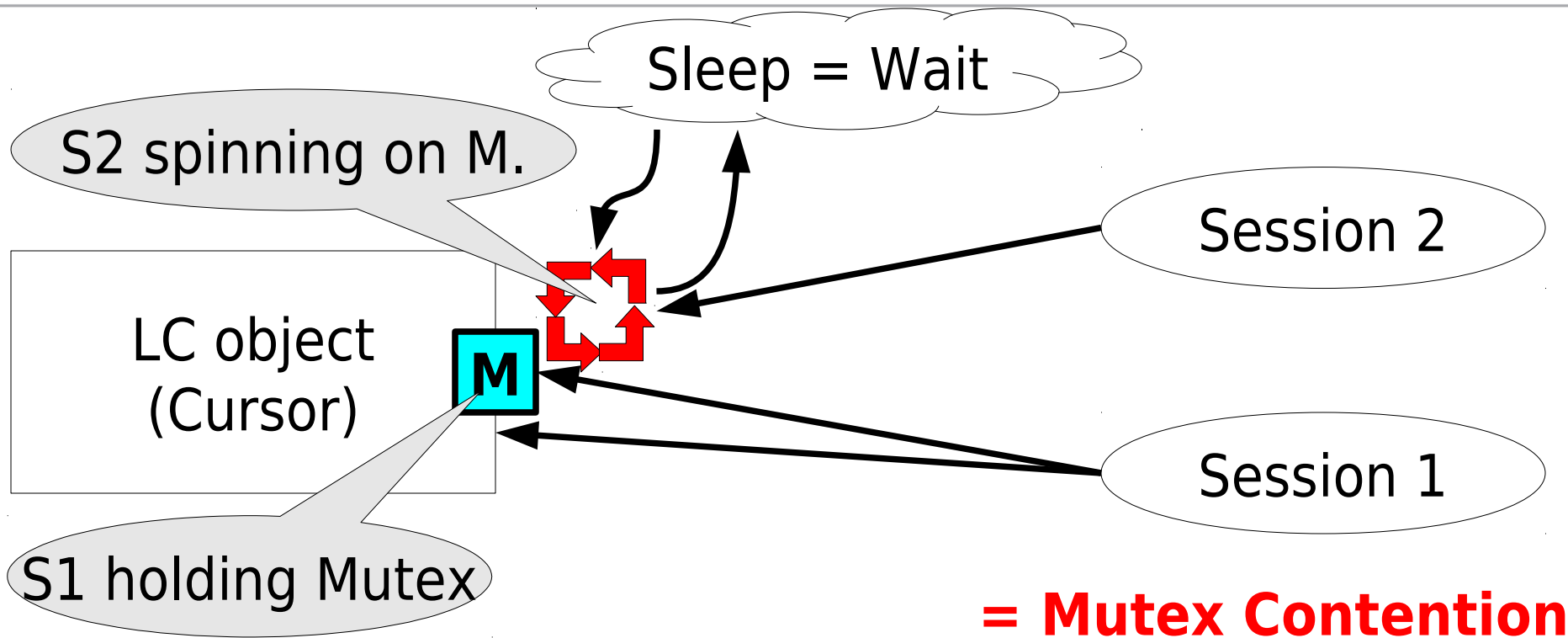
(picture from Wikipedia)

Mutex



- “Mutual Exclusion”
= Fine-grained serialization structure
- It's just like a **latch**, but smaller, lighter, **faster**
- Introduced in 10g R2
- managed by KGX (Kernel Generic Mutex Module)

Waits



What are Wait Events (on Mutexes)?

- Somebody **requests** a Mutex
- Can **not get it** by spinning
- And thus, **sleeps**

- Sleeping is recorded as wait time
- Spinning is **not** recorded as wait-, but as CPU time

cursor: mutex X

Wants: **Exclusive** mode on Mutex of Parent / Child

To:

- Build a new Child Cursor
- Capture SQL bind data (peek)
- Modify cursor-related statistics

cursor: mutex S

Wants: **Shared** mode on Mutex of Parent / Child

To:

- Change the reference count (“in flux”)
= “new guy is interested / spinning”

cursor: pin X

Wants: **Exclusively** pin a P/C cursor in cache

To:

- Create the cursor
- Alter the cursor

cursor: pin S

Wants: Pin a P/C cursor in **shared** mode

To:

- Use (execute) the cursor


cursor: pin S wait on X

Wants: Pin a P/C cursor in **shared** mode
but **sb. already has it** in **exclusive** mode

To:

- Use (execute) the cursor
- When sb. is altering the cursor (e.g. due to DDL)

Agenda

- Introduction
- Oracle Parsing, Child Cursors
- Mutexes, Waits and Reasons
- Issues, Quick Fixes and Solutions 
- Summary, Acknowledgements, Q&A session

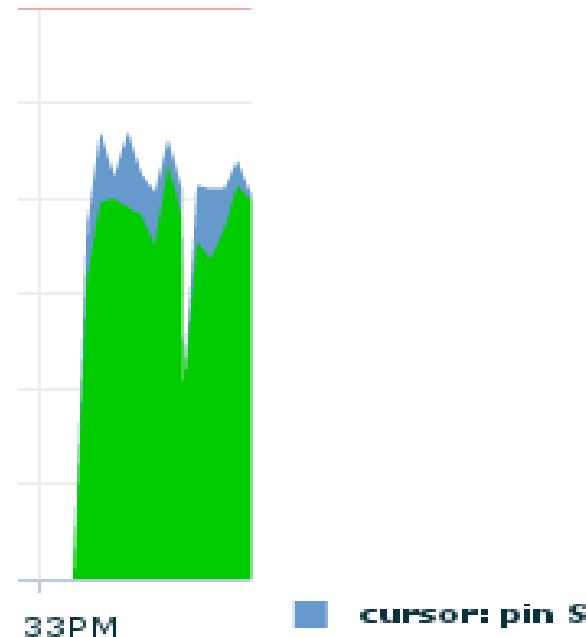
Simple - cursor: pin S

- Caused by massively parsing one SQL_ID
=> Hot Spot Object in Library Cache
- Diagnosis: Oracle Wait Interface
- (Half) solution: Diversify SQL_ID (not **randomize!**)
`select /* WebServer4 */ something from table;`

Provocation of a cursor: pin S wait situation

- Tightly looping one SQL
1,000,000 times
- in 20 threads

→ Overcrowding the cursor's pin mutex



Complex - cursor: mutex S/X

- Root-caused by invalidated child cursor(s)
=> Too many cursor objects in Library Cache
- Diagnosis:
 - Oracle Wait Interface
 - 10046 Level 12 session trace (=sql_trace event)
 - v\$sql_shared_cursor **plus** cursortrace [296377.1]

One example reason for cursor: mutex S/X

Application uses jdbc setter methods
improperly on INTEGER column (=2)

- setNUMBER(2) => Bind Var. is **NUMBER**
- setNULL(2) => Bind Var. is **VARCHAR2**

= BIND MISMATCH

10046 Level 12 trace for cursor: mutex S/X

Trace 1:

Bind#2

```
>> oacdty=01 mxl=32(04) mxlc=00 mal=00 scl=00 pre=00  
oacflg=03 fl2=1000010 frm=01 csi=873 siz=0 off=168  
kxsbbbfp=1118e1cd8 bln=32 avl=00 flg=01
```

Trace 2:

Bind#2

```
>> oacdty=02 mxl=22(22) mxlc=00 mal=00 scl=00 pre=00  
oacflg=03 fl2=1000000 frm=01 csi=873 siz=0 off=168  
kxsbbbfp=110977db8 bln=22 avl=02 flg=01  
value=99
```

**in 30 columns
= 2³⁰ times
BIND MISMATCH**

One Quick Fix for cursor: mutex S/X

System is loaded with heavy mutex waits
due to high number of cursors (=version count)

=> **frequently** flush this cursor with
dbms_shared_pool.purge
(look out for new parsing issues = CPU)

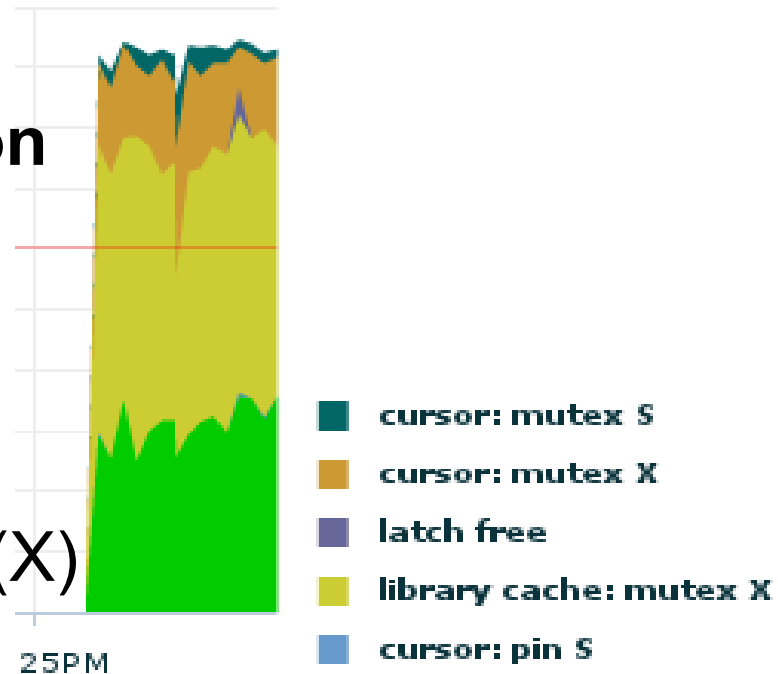
One solution for cursor: mutex S/X

Application uses jdbc setter methods
now properly on INTEGER column (=2)

- `setNUMBER(2)` => Bind Var. is **NUMBER**
- `setNULL(2, java.sql.Types.INTEGER)`
 => Bind Var. is **NUMBER**

Provocation of a cursor: mutex S/X wait situation

- Generating 64 child cursors for one SQL_ID
- Accessing them 20x parallel
- Delay to create new children (X)
- Delay to select good child (S)



Similar Problem with CHAR binds

- Bind buffers are size 32, 128, 2000 or 4000 bytes
- Changing CHAR bind length invalidates
- Reason BIND_LENGTH_UPGRADEABLE
= 4^n cursor versions

Heavy - Oracle internal pitfalls I

- 11g “features” like

MOS: “Its important to note that cursor obsoletion code was removed in version 11.

*That means **we no longer obsolete a parent cursor when it reaches 1024 child cursors** [as we did in 10g.]”*


- Workaround

*Enhancement Patch 10187168 introduces parameter
“_cursor_obsolete_threshold”*

Heavy - Oracle internal pitfalls II

- DB Bugs like 10157392 + 12939876
(fixed in 12.1, backported to 11.2.0.3)
Memory leak: increasing number of child cursors over time, especially if the shared pool is under load.
- DB Bug 9591812 (fixed in 12.1)
Wrong wait events in 11.2 ("cursor: mutex S" instead of "cursor: mutex X")
Official MOS workaround:
Be cautious when interpreting S mode mutex / pin waits....

Agenda

- Introduction
- Oracle Parsing, Child Cursors
- Mutexes, Waits and Reasons
- Issues, Quick Fixes and Solutions
- Summary, Acknowledgements, Q&A session 

I suggest...

My suggestions for “cursor: mutex S/X” casualties

- Check how the application does handle **bind** variables
Avoid BIND MISMATCH at (nearly) any cost
- Reduce the number of **cursor versions** below 100
More will lead to overhead
- Look for matching Oracle **bugs** in your RDBMS release
- **Upgrade** to 11.2.0.3 or higher
11.2.0.2 is worst version for cursor issues IMHO

More resources on this topic

- MOS Documents
1356828.1; 1377998.1; 296377.1
- Pöder, Tanel
Presentation: “Oracle Latch and Mutex Contention Troubleshooting”
- Shallahamer, Craig
Book: “Oracle Performance Firefighting”
(ISBN 978-0-9841023-0-3)
- Nikolaev, Andrey
Blog entries: “Mutex waits. Part 1 + 2”



Oracle OpenWorld
September 30 - October 4, 2012
San Francisco



Thank you very much for your attention!

Martin Klier
Senior DBA
Klug GmbH integrierte Systeme

San Francisco, Sept 30th, 2012



Thank you

Many people have helped with suggestions, supplying test cases or taking daily work off me during preparation and travel phase. Guys, you are top!

My special thanks to:

My boss and company, for endorsement

Two special customers, for interested patience

An ex-colleague, for The Code

One guy from OR, who made it possible.