



Changing the rules of business

Distributed Data Processing with ILOG JRules



UBS Investment
Bank

Daniel Selman
Architect, ILOG

Keith Lindsey
Associate Director, UBS

- Business Problem
- Distributed Architecture
- Choosing an Execution Algorithm
- Sample Rules
- Sequential Mode
- Results

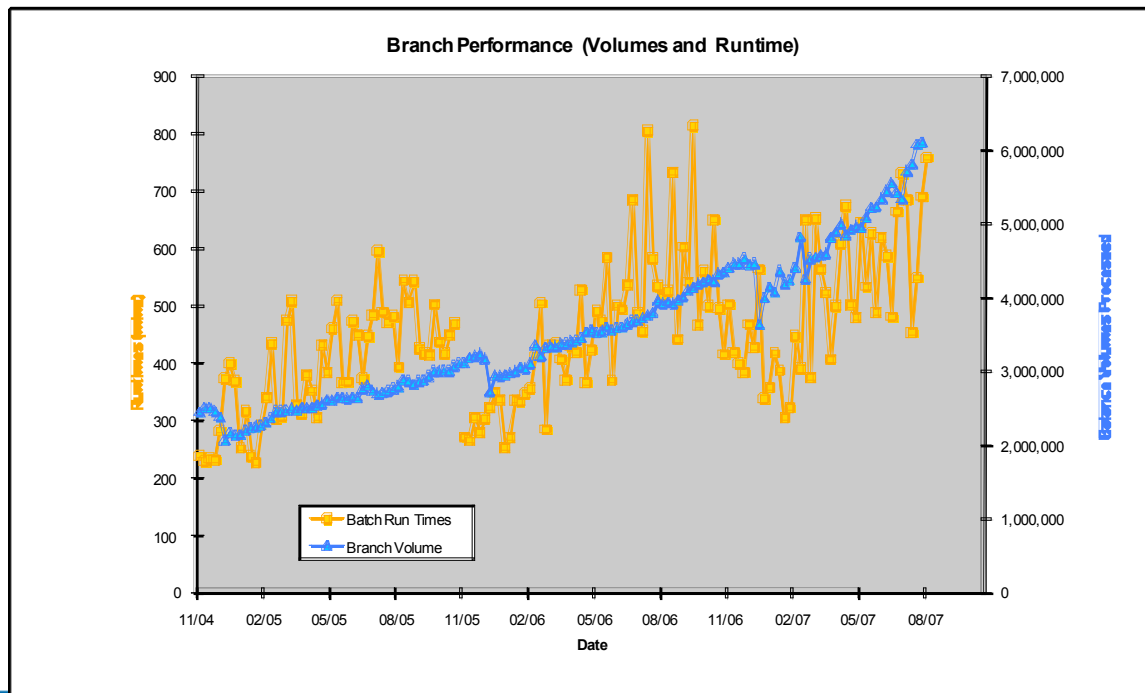
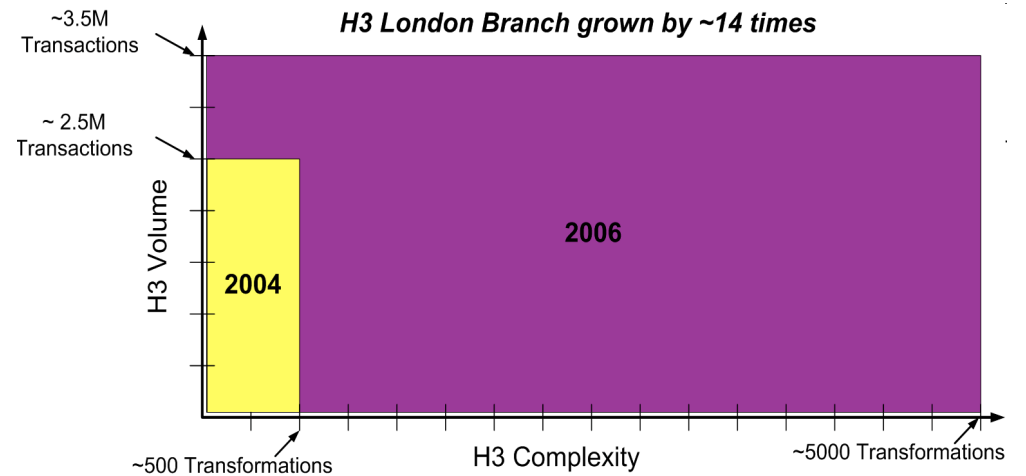
- **UBS Bank operates in over 50 countries**
- **Transform the General Ledger Financial Details Database into Regulatory and Internal Reports**
 - Required by 8AM for the banks in all geographies
 - Reports go to corporate centre as well as local banking regulators
 - Support different accounting standards
- **Legacy application based on ETL was too hard to maintain and did not meet business or performance requirements**

Performance and Scalability



Changing the rules of business

- Complexity and volume have grown beyond the original design capabilities
- Was not meeting the business requirements for timeliness
- Transformation rules were complex to maintain, run and configure



- Should be able to scale horizontally and vertically – distribute the data across numerous fast and low cost boxes
 - Distribution means less data to process on each box, hence faster processing
 - Keep data close to code for faster processing – do not drop data unless required
 - No time wasted storing data that you do not want
 - By regulating the amount of data of each box it will be possible stabilise performance as volume grows

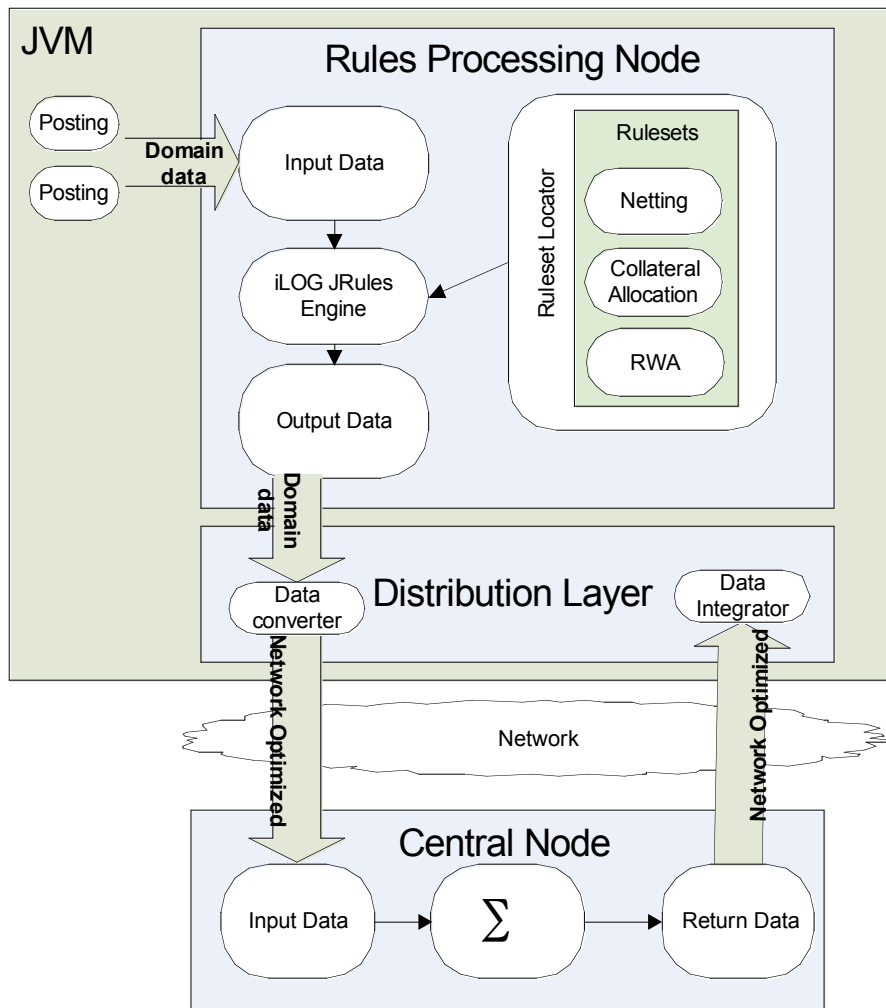
- Applications are often classified according to how often their subtasks need to synchronize or communicate with each other.
 - An application exhibits fine-grained parallelism if its subtasks must communicate many times per second;
 - It exhibits coarse-grained parallelism if they do not communicate many times per second,
 - And it is embarrassingly parallel if they rarely or never have to communicate.

Wikipedia

Rules in the Distribution Framework



Changing the rules of business

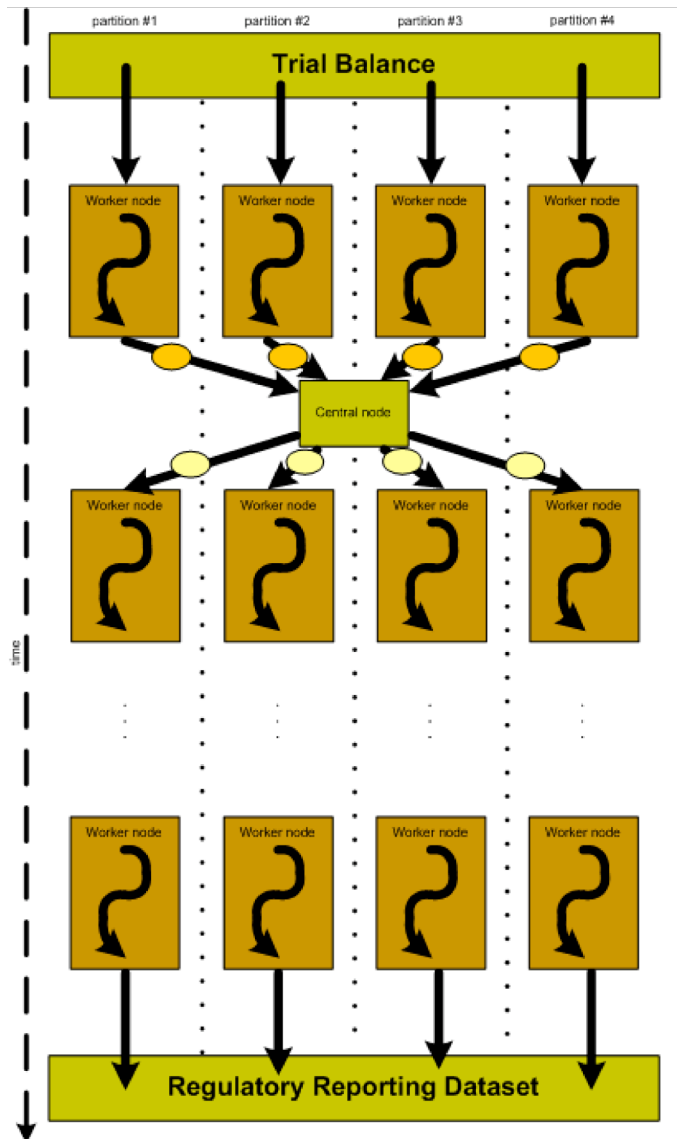


- Business logic is coded in rules
- When a sum/sort is needed, data is sent to central node and the result is returned
- The rules only interact with the domain data objects, not the data transfer objects
- This keeps the rules abstracted from the distribution framework they run in

Implementation Summary



Changing the rules of business



- Partition dataset into segments
- Process each segment in parallel using a grid of worker servers
- Each worker communicates with a central server using a lightweight protocol
- Synchronization points with the central server are kept to an absolute minimum
- Rules process data held in-memory

- **Compliance and Validation**
 - Loosely interrelated rules that check a set of conditions to yield a go/no go or similar constrained result. Compliance business rule applications are often used in underwriting, fraud detection, data validation and form validation. The business rules in this kind of application would generally have a yes/no result and would provide some explanation on the decision.
- => **Sequential or Fastpath**

- Computation

- Strongly interrelated rules that compute metrics for a complex object model. Computation business rule applications are often used for scoring and rating, contracts and allocation. The business rules in this kind of application would carry out different calculations on an object which would be responsible for providing a final value (or rating).

- => RetePlus

- Correlation
 - Strongly interrelated rules that correlate information from a set of objects, to compute some complex metrics. Correlation business rule applications are often used for billing. The business rules in this kind of application insert information.
- => RetePlus or Fastpath

- **Stateful Session**
 - Strongly interrelated rules that correlate events in a stateful engine session. Stateful applications are often used in alarm filtering and correlation, GUI customization, and Web page navigation.
- => **RetePlus**

JRules Execution Modes



Changing the rules of business

In your rule task:	RetePlus	Sequential	Fastpath
Compliance and validation application	✗	✓	✓
Computation application	✓	✗	✗
Correlation application	✓	✗	✓
Stateful application	✓	✗	✗
Working memory objects	✓	✗	✓
Rule chaining	✓	✗	✗
Tests on existence or collection items directly in working memory	✓	✗	✓
Shared test patterns	✗	✗	✓
Heterogeneous bindings	✓	✗	✓
Dynamic priorities	✓	✗	✗
Runtime rule selection that selects a few rules among many	✓	✓	✗ ¹
Numerous rules	✗	✓	✓

E.g.
Decision
Tables

Rule1	... when{A();B()} ...
Rule2	... when{A()} ...
Rule3	... when{B()} ...

Rule1	... when{Person(age == 12);} ...
Rule2	... when{Person(age > 20);} ...

- **Statistics**
 - If-then-else BRL rules: 749
 - Decision tables: 55
 - Decision trees: 5
 - Ruleflows: 111

Sample Rule 1



Changing the rules of business

definitions

set 'the posting' to a Posting in the postings filtered by product types of 'the internal data' ;

set 'the trade balance' to a Trade Balance from the Trade Balance of 'the posting' ;

set 'the Reporting Date' to the Reporting Date ;

if

Number of days between 'the Reporting Date' and the Settlement Date (TB115) of 'the trade balance' is less than 0

then

add posting 'the posting' to the open trade postings list ;

else

add posting 'the posting' to the failed trade postings list ;

Sample Rule 2



Changing the rules of business

definitions

set 'the posting' to a Posting in the list of input postings where this Posting is selected for PP50 processing ;

set 'the GCR entity' to the gcr of 'the posting' ;

set 'the counterparty' to the counterparty of 'the posting' ;

set 'the parent' to the Parent Counterparty of 'the counterparty' ;

set 'the country of ultimate risk' to the Country of Ultimate Risk (PA060) of 'the parent' ;

set 'the code of the country' to the Country Code of 'the country of ultimate risk

if

the ultimate risk country code of PP50 GCR Input Table Result of 'the posting' equals 3

then

set the ultimate risk country code of 'the GCR entity' to 'the code of the country'

Sample Rule 3



Changing the rules of business

definitions

set 'the posting' to a Posting in the list of postings;
set 'the lookup result' to the lookup results for 'the posting' in 'PP72 Input Table Lookup' ;
set 'the residual maturity' to the selected residual maturity of 'the lookup result' ;
set 'the original maturity' to the selected original maturity of 'the lookup result' ;
set 'the product type group' to the product type group of 'the lookup result' ;

if

all of the following conditions are true :

- 'the lookup result' is not empty is true
- (the Transaction Netting ID (PO998) of 'the posting' is not "N" and
the Transaction Netting ID (PO998) of 'the posting' is NOT EMPTY)
- ('the product type group' is "RV" and
the Notional Exclusion Flag (PO995) of 'the posting' is not N) is false

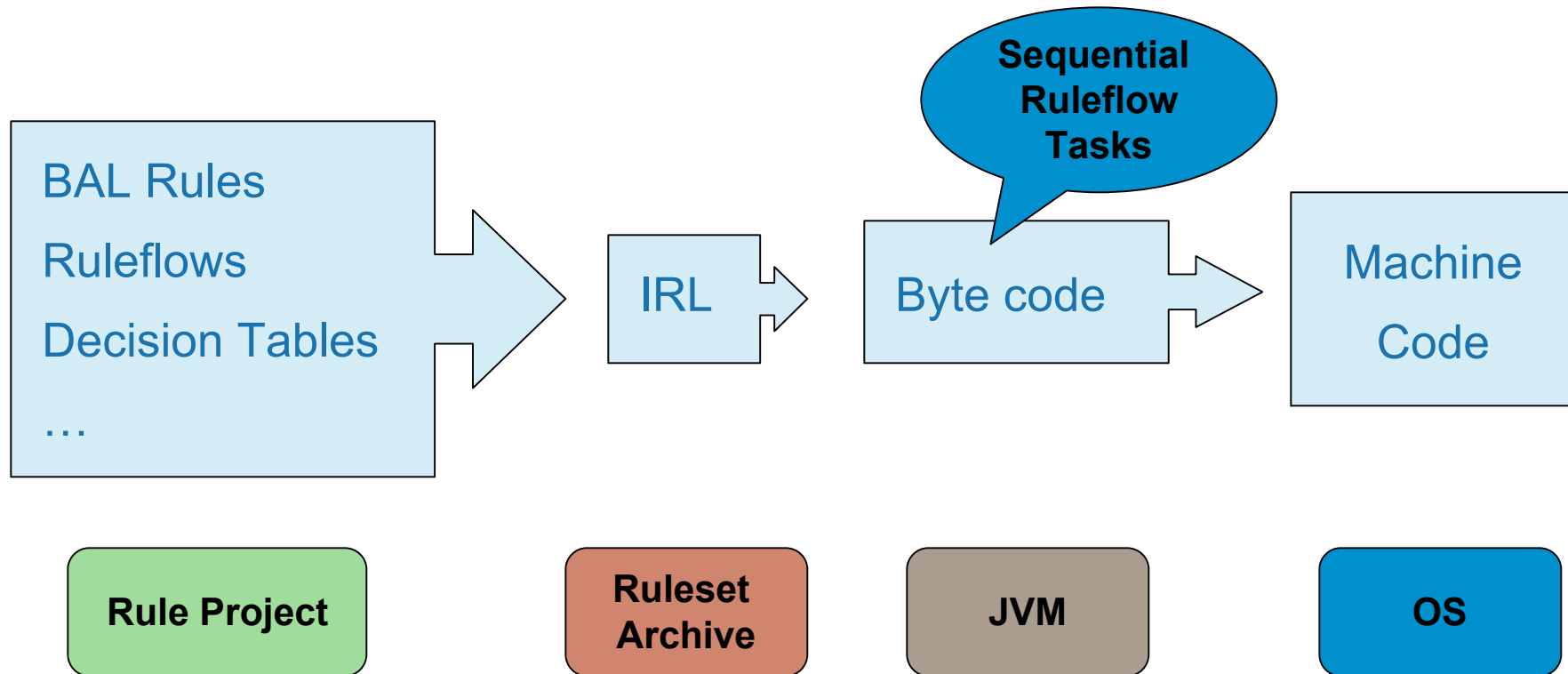
then

set the product type group of 'the posting' to 'the product type group';
set the selected residual maturity of 'the posting' to 'the residual maturity' ;
set the selected original maturity of 'the posting' to 'the original maturity' ;
group 'the posting' ;

Sequential Mode



Changing the rules of business



definitions

set 'the posting' to a Posting in the postings filtered by product types of 'the internal data' ;

set 'the trade balance' to a Trade Balance from the Trade Balance of 'the posting' ;

set 'the Reporting Date' to the Reporting Date ;

if

Number of days between 'the Reporting Date' and the Settlement Date (TB115) of 'the trade balance' is less than 0

then

add posting 'the posting' to the open trade postings list ;

else

add posting 'the posting' to the failed trade postings list ;

```
rule ClassifyOpenAndFailedTradePosting {
    property status = "new";
    when {
        com.ubs.meridian.internaldata.PP30InternalData() from internalData;
        the_posting: com.ubs.meridian.data.interfaces.Posting() in
        internalData.postingsFilteredByProductType;
        the_trade_balance: com.ubs.meridian.data.interfaces.TradeBalance() from
        the_posting.tradeBalance;
        the_Reporting_Date: com.ubs.meridian.engine.types.CalendarDate() from
        com.ubs.meridian.inputdata.PP30InputData.vgetLBD();
        evaluate (the_Reporting_Date.numberofDaysBetween(the_trade_balance.settlementDate) <
        0);
    } then {
        com.ubs.meridian.internaldata.PP30InternalData.vAddToOpenTradePostingsList(the_posting);
    } else {
        com.ubs.meridian.internaldata.PP30InternalData.vAddToFailedTradePostingsList(the_posting);
    }
}
```

Pseudo (Byte) Code



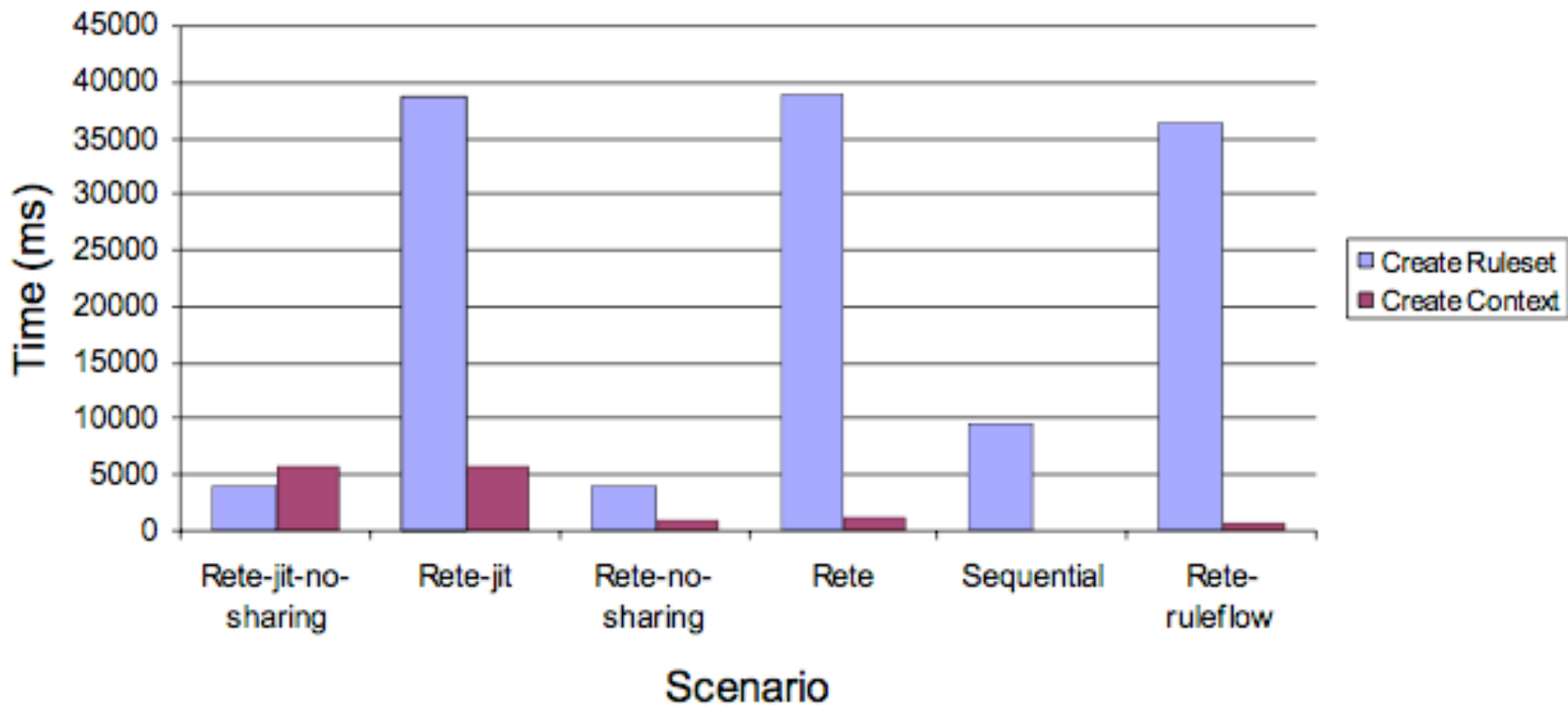
Changing the rules of business

```
protected final void rule_ClassisfyPosting_ClassifyOpenAndFailedTradePosting()
{
    com.ubs.meridian.internaldata.PP30InternalData __V0 =
    (com.ubs.meridian.internaldata.PP30InternalData)this.globalDriver.getGlobalObject(4,7);
    if (__V0 != null)
    {
        java.util.List __V1 =
        ((com.ubs.meridian.internaldata.PP30InternalData)this.globalDriver.getGlobalObject(4,7)).getPostingsFilteredByProductType();
        if (__V1 != null)
            if (__V2 != null)
                if (__V2 instanceof com.ubs.meridian.data.interfaces.Posting)
                {
                    com.ubs.meridian.data.interfaces.Posting the_posting = (com.ubs.meridian.data.interfaces.Posting)__V2;
                    {
                        com.ubs.meridian.data.interfaces.TradeBalance the_trade_balance = the_posting.getTradeBalance();
                        if (the_trade_balance != null)
                            {
                                com.ubs.meridian.engine.types.CalendarDate the_Reporting_Date =
                                function_translation_com_ubs_meridian_inputdata_PP30InputData_vgetLBD();
                                if (the_Reporting_Date != null)
                                    if (the_Reporting_Date.numberofDaysBetween(the_trade_balance.getSettlementDate()) < 0)
                                        {
                                            function_translation_com_ubs_meridian_internaldata_PP30InternalData_vAddToOpenTradePostingsList(the_posting); }
                                        else
                                        {
                                            function_translation_com_ubs_meridian_internaldata_PP30InternalData_vAddToFailedTradePostingsList(the_posting);
                                        }
                                    }
                                }
                            }
                    }
                }
            }
    }
}
```

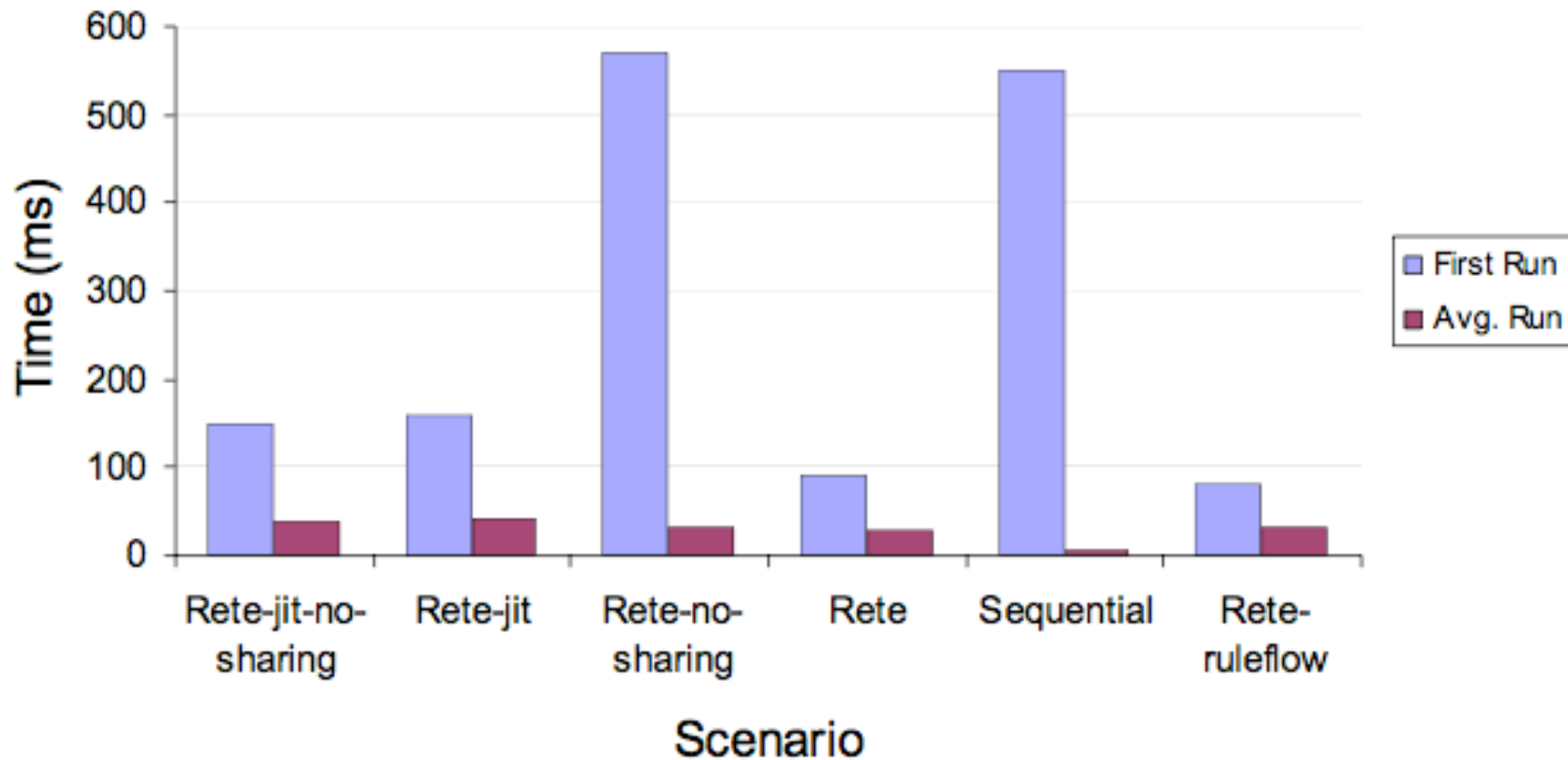
- Limitations

- *not, exists, collect* conditions without an enumerator (as no working memory)
- Lesser used features
 - dynamic priorities (as no agenda)
 - event-based condition
 - Watchdogs
 - TMS

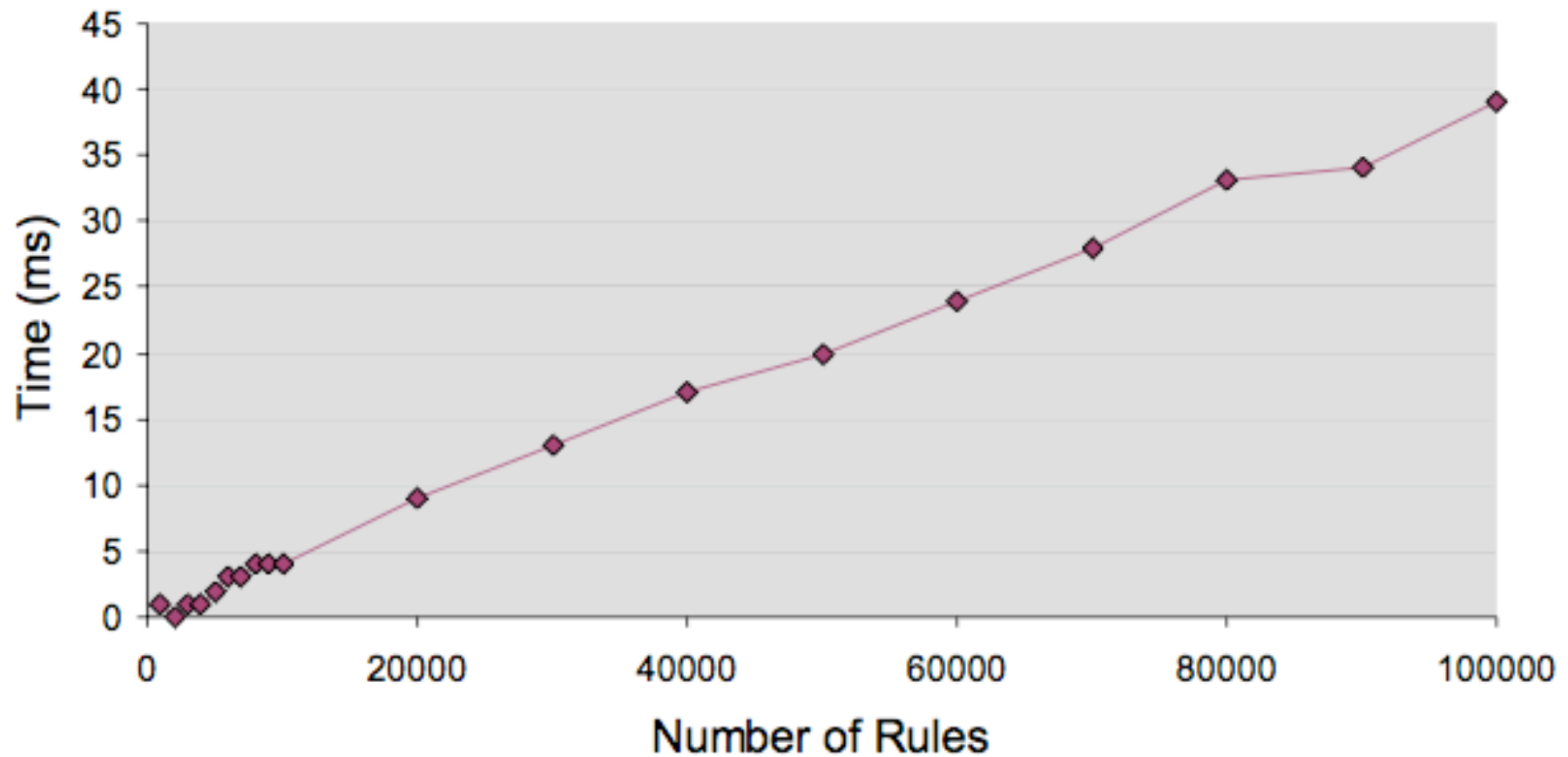
10,000 Rules - Setup



10,000 Rules - Execution



Average Run, Sequential Algorithm

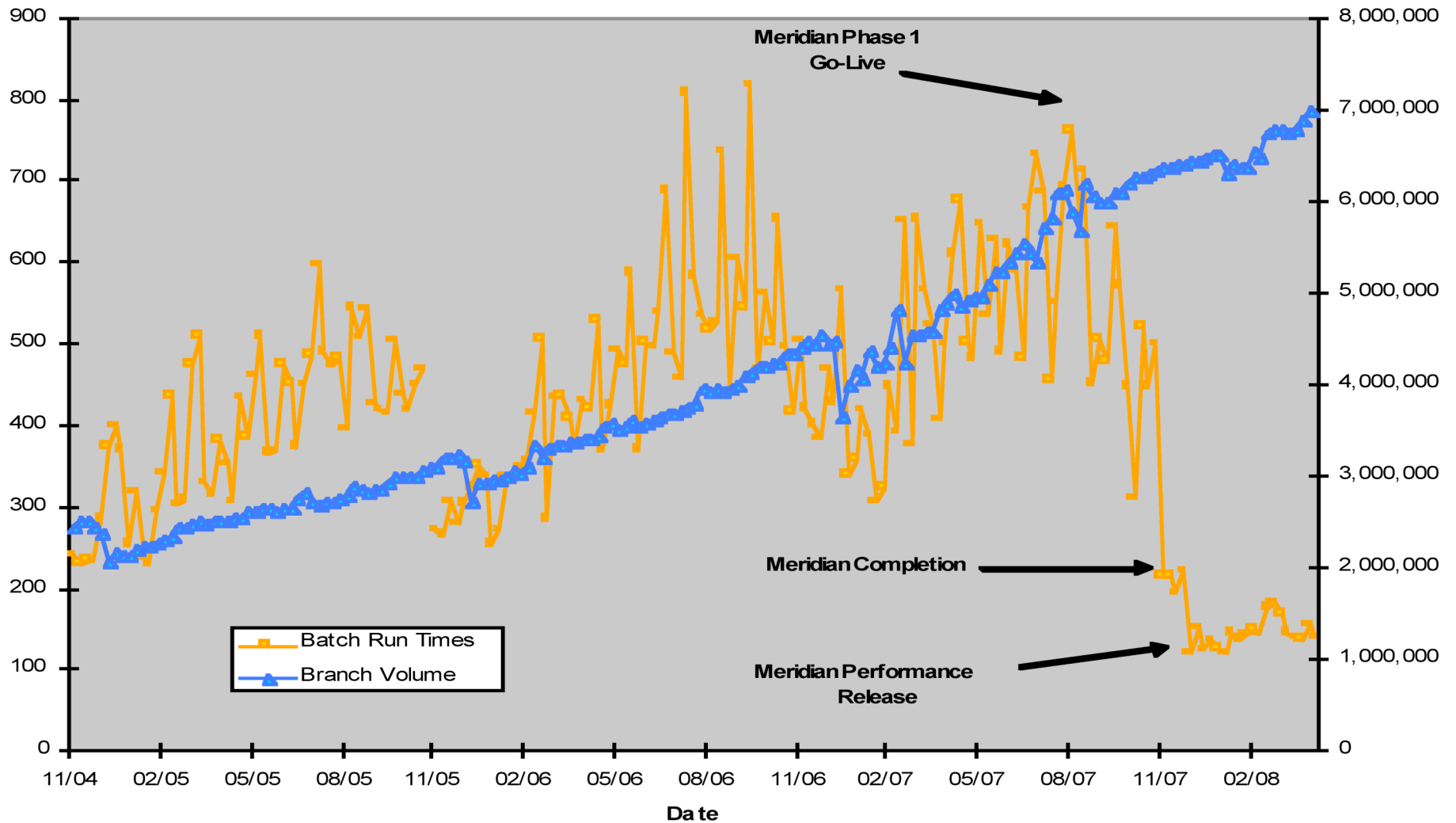


Results: UBS Performance



Changing the rules of business

Branch Performance (Volumes and Runtime)



- While data volumes have been steadily increasing, dramatic runtime reductions have been achieved since the project went live
- Regulatory reporting data is now predictably delivered (very low runtime variance)
- Increased agility allows for faster time to market in terms of business change
- Common vocabulary now in place between the business and technical teams



dselman@ilog.fr