UNIVERSITY FINANCE SEMINAR 14 MARCH 2003

Current Alternative Approaches to Formal Contract Representation

M A H Dempster

Centre for Financial Research

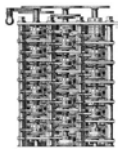
Judge Institute of Management, University of Cambridge

and

Cambridge Systems Associates Limited

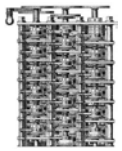
mahd2@cam.ac.uk **http://www-cfr.jims.cam.ac.uk**

Co-workers: H G Go, I N Larkin & E A Medova



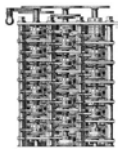
Outline

- Introduction and Problem
- Solution Space
  - Product standards
  - Valuation systems
  - Formal languages
- Conclusions

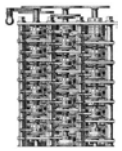


Contract Representation in Financial Institutions

- Derivatives volume is still expanding exponentially with exchange traded instrument business growing and OTC business slowing in some areas and increasing very rapidly in others such as credit derivatives
- Derivatives desks are making an increasing contribution to banks' bottom lines
- Term sheets are currently interpreted separately by front, middle and back office
- Most banks have significantly reduced the number of operational systems in recent years
- In many asset class areas individual pricing DLLs are still in widespread use
- Most leading institutions have projects underway to consolidate pricing and risk management on distributed architectures with thin client access

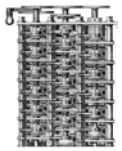


- Cross asset coverage is however very limited at this point
- Although consistency of trade representation over its lifetime is still mostly based on term sheets a number of institutions are investigating XML, FpML, etc. for formal contract representation and trade confirmation
- Some banks are thinking about the formal structure of complex contracts in specific asset areas where the focus is always on the latest instruments
- Term sheets seem to be still the usual way of defining risk hot spots
- Control of the operational risk of the STP of trades is still based on business rather than IT processes
- Many banks see FpML based interbank messaging for high volume contracts as the future although most currently use traditional methods



The Objective

- *Concise unambiguous* and *complete* representation of complex financial contracts primarily for intrabank but also for interbank communication
  - Aid efficient communication between multiple user groups
    - Front office: traders, quants…
    - Middle office: product control, derivatives IT,…
    - Back office: risk management, settlement, accounting,…
  - Reduce operational risk from coding errors, misunderstandings, etc
  - Infrastuctural support systems
    - Integration with existing databases / datafeeds
    - Integration with existing valuation tools / analytics
    - Risk management (e.g.VaR calculation)
    - Documentation
  - Minimize overhead and maximize component reuse
  - Possibility of symbolic computation on contracts

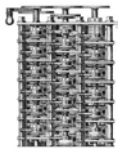


Status Quo

- Natural language (English) term sheets and confirmations which may refer to master agreements and ISDA definitions

- Spreadsheets/VBA
  - Free form (allows expression of anything) but no semantic structure so tools must be reproduced on case-by-case basis
  - Very poor data validation/data protection (see http://panko.cba.hawaii.edu/ssr)
- Databases
  - relational databases designed for holding many instances of the same structure
  - object-oriented databases for flexible structures

- Proprietary systems (e.g. Sungard Panorama)

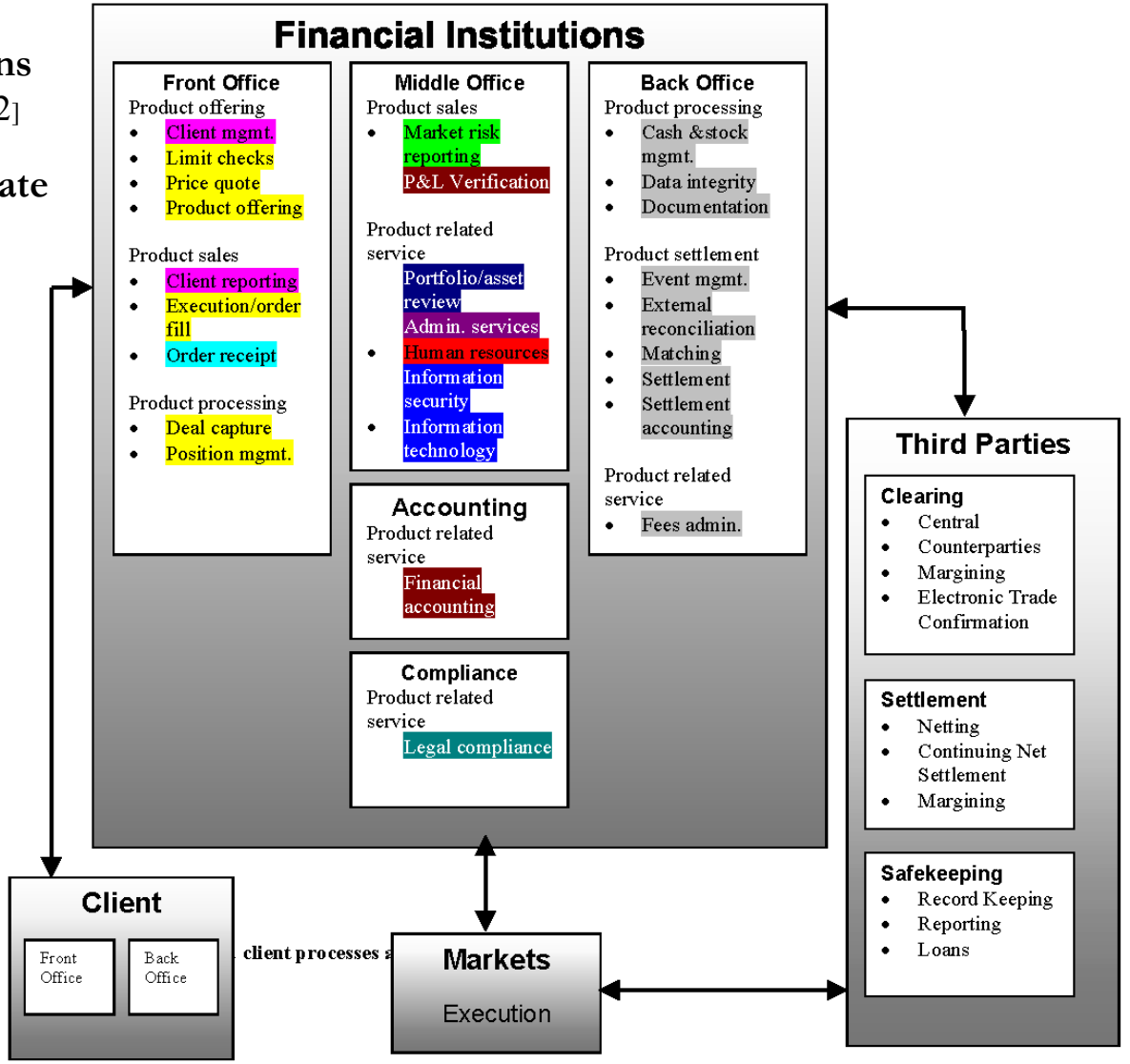


Intrafirm and client processes and connections
Meridien Research [GARP Risk Review, Issue 6, 2002]

Example of processes necessary for interest rate derivatives
Dresdner Bank [RISK, January 2003]

Color-coding represents processes performed within following business units

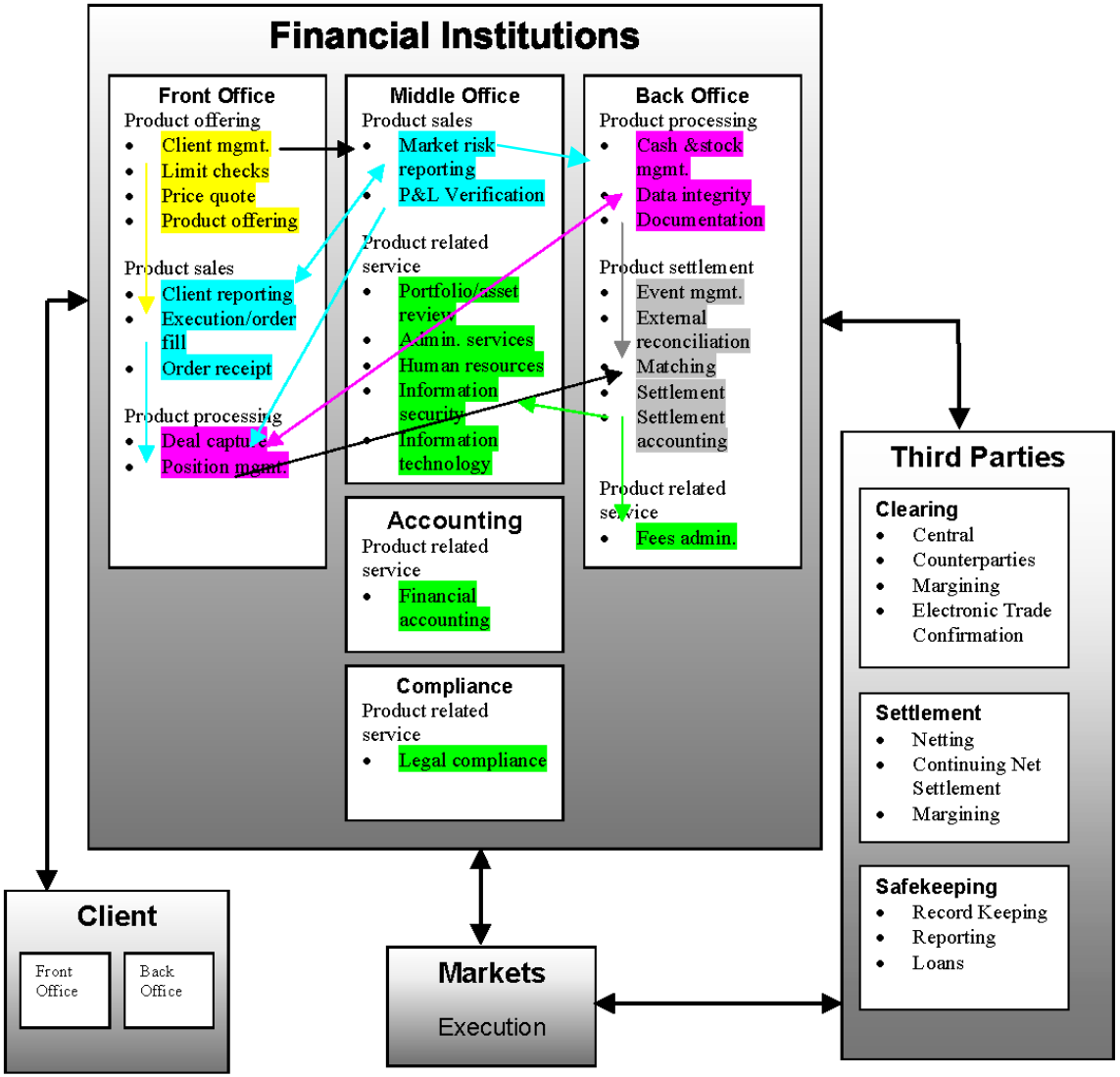
- Sales & Marketing
- Rates
- Short Term Products/Treasury
- Corporate Center
- Finance
- Credit Review
- Operations
- Admin. Services
- Human Resources
- Information Technology
- Compliance
- Credit Risk

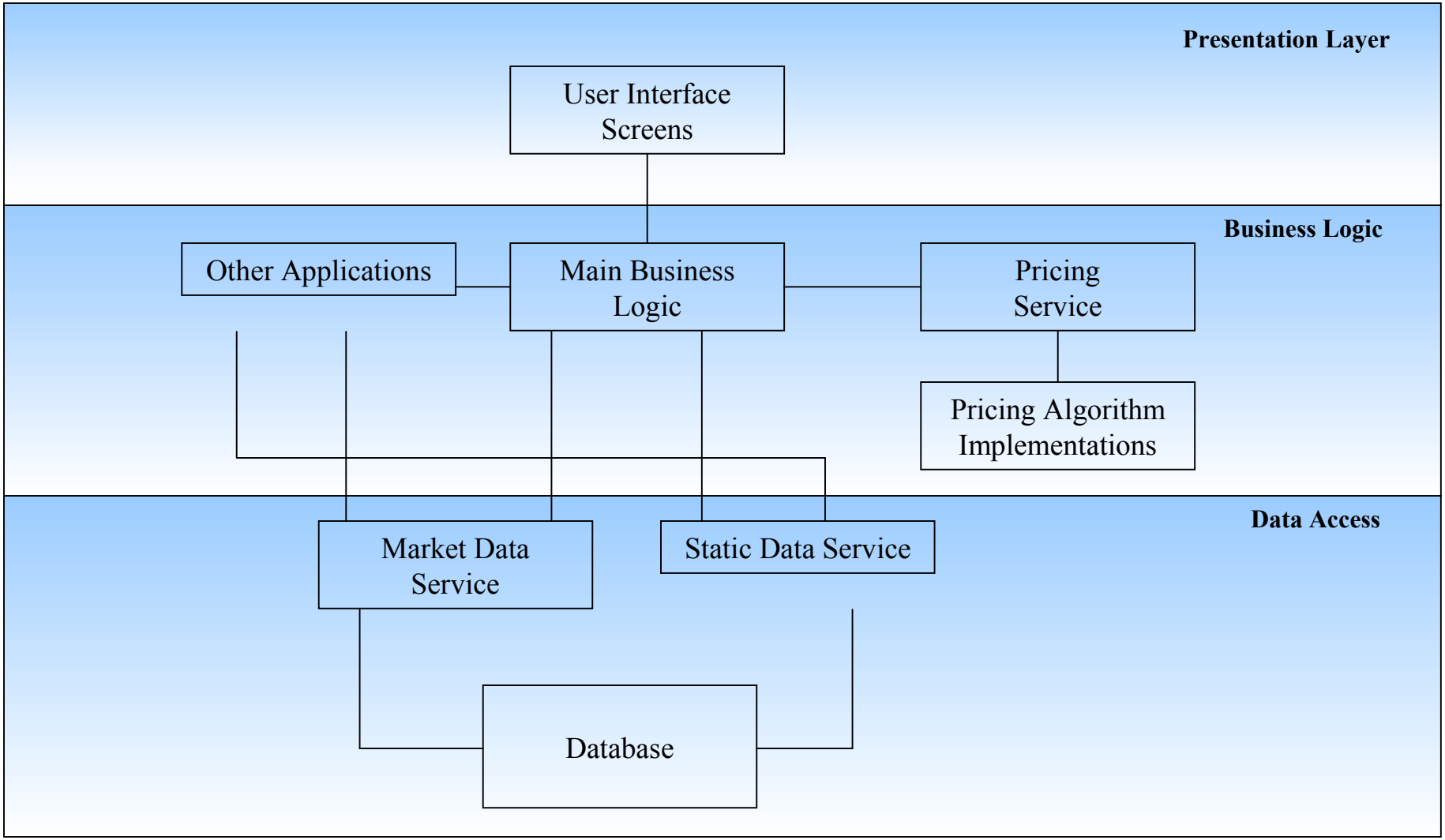


Bank inter-communication
Example of processes necessary for interest rate derivatives
Dresdner Bank [RISK, January 2003]

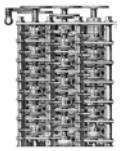
Color-coding represents the stages in product life grouped according to Dresdner Bank's Risk Scorecard Approach

Product offering
Product sales
Product processing
Product settlement
Product related service



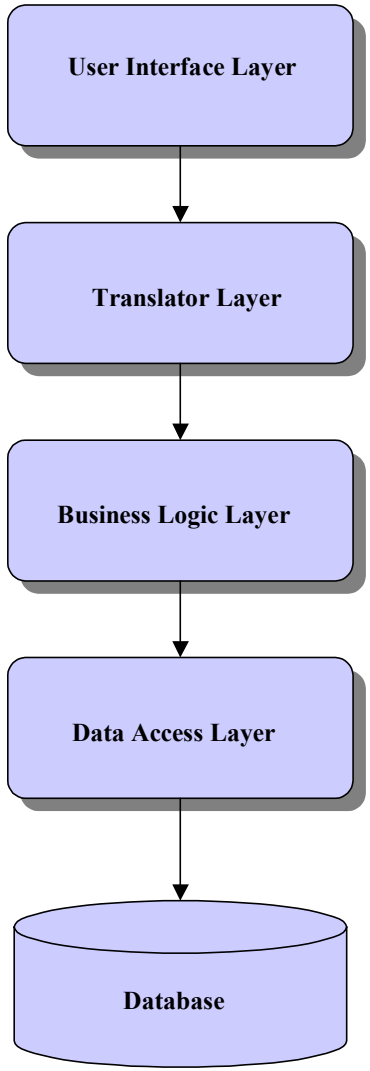


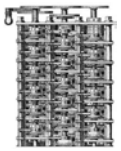
Architecture of an n-layer (thin client) trading and risk management system
Source: Overhaus *et al* (2002)



Business logic – pricing algorithm code – shielded by a layer that maps SOAP messages into function calls with market or stored data access provided by another web services.

Source: Overhaus *et al* (2002)

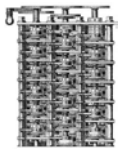




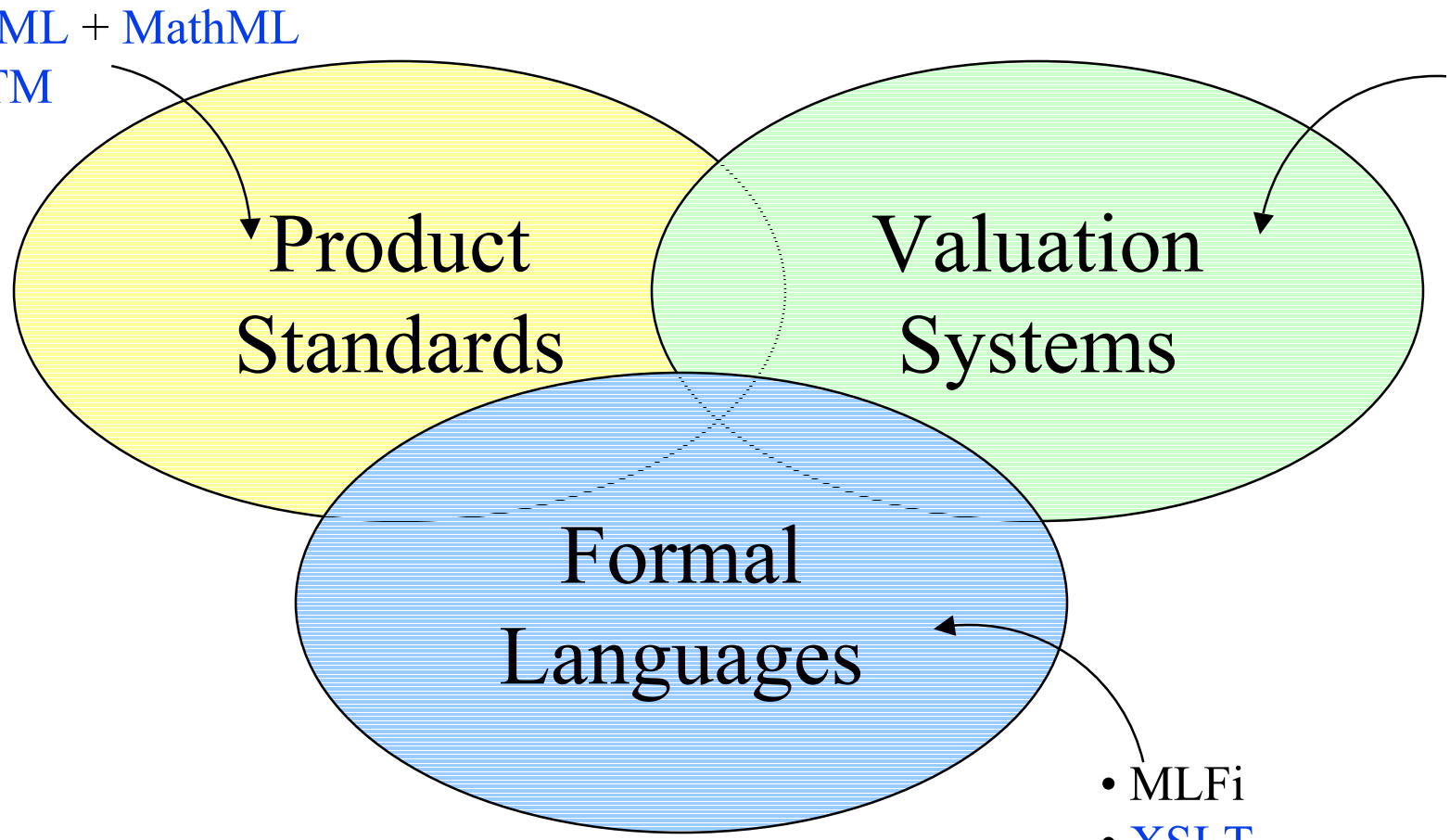
The Problem

- Provide a concise unambiguous and complete representation of complex financial contracts

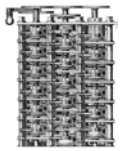
- Allow for the possibility of symbolic computation on contracts in order to provide formal verification of product structure and automatic documentation including term sheets

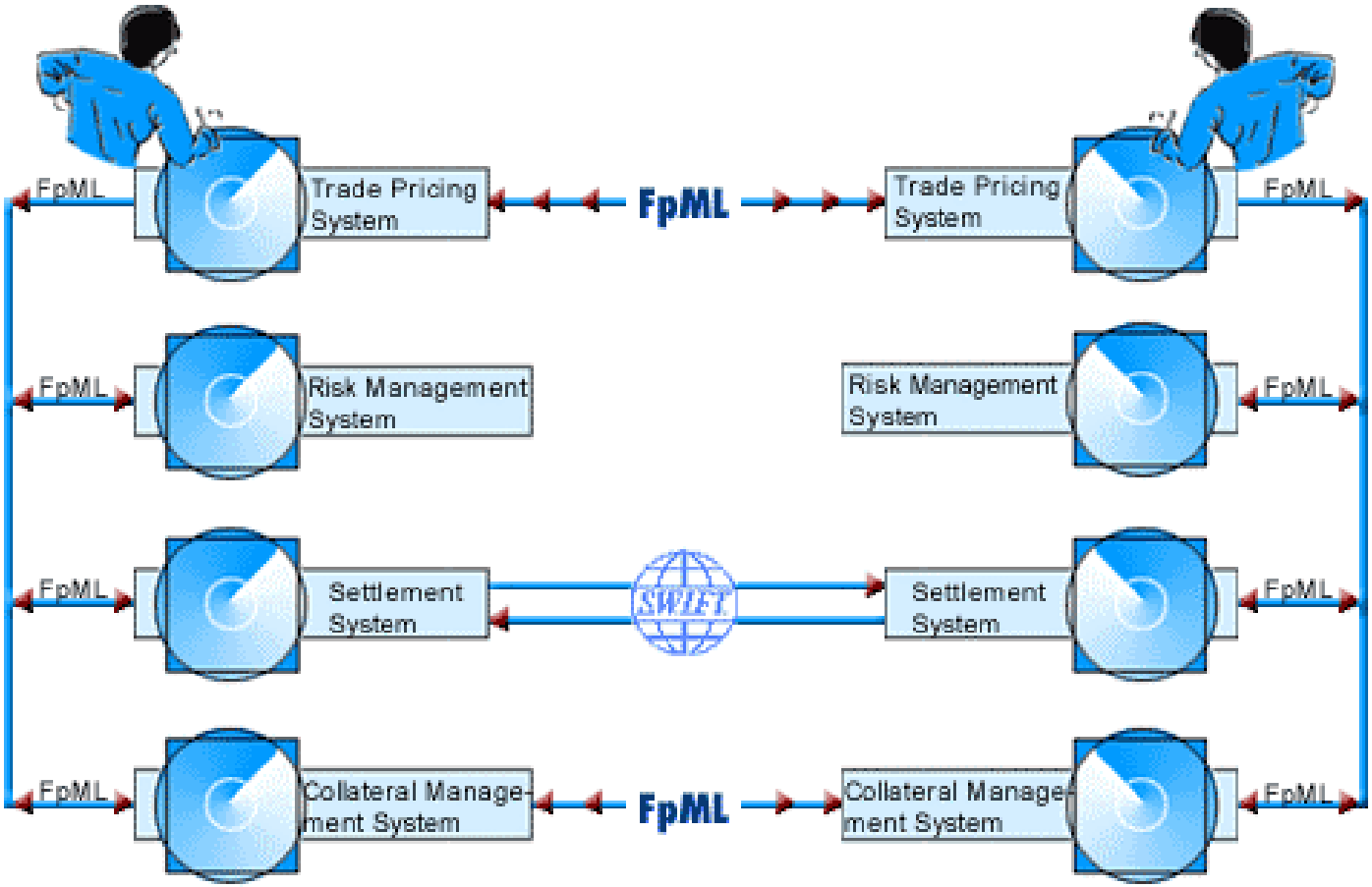


Solution Space



XML Based

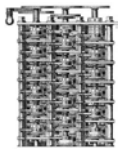




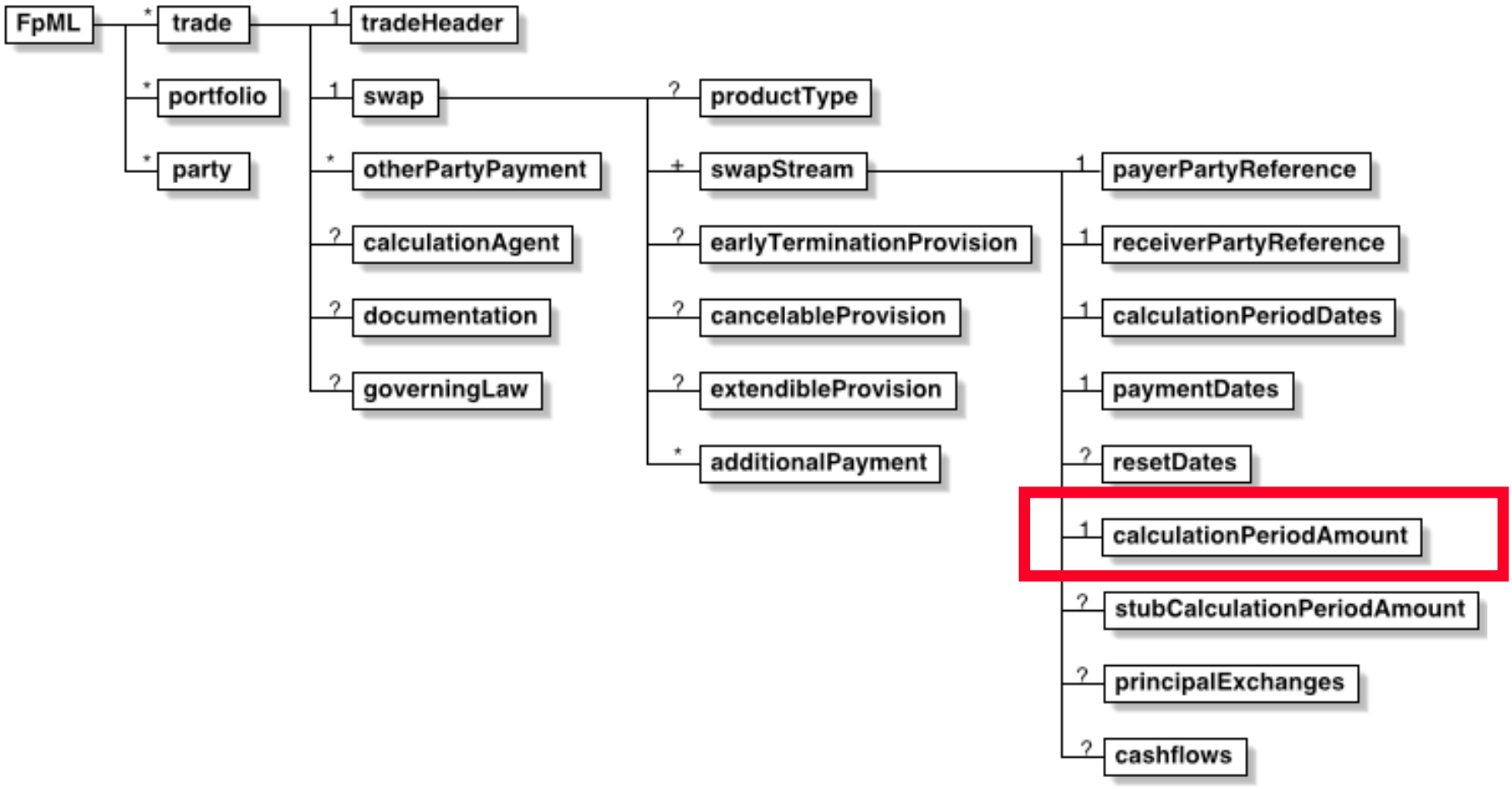
FpML as an interbank standard. **Source**: **www.fpml.org**

Financial Products Markup Language

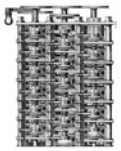
- XML protocol for e-commerce of financial derivatives
  - Controlled by FpML standards committee www.fpml.org
  - Eventually all types of OTC derivatives
  - Electronic integration across trading and risk management
- V3.0 latest Working Draft publicly available
  - Standard instrument specifications
  - Vanilla FX & IR derivatives, e.g. FRA's and swaps
- V4.0 in discussion within committee
  - Vanilla equity options
- Trailing behind product innovation by banks



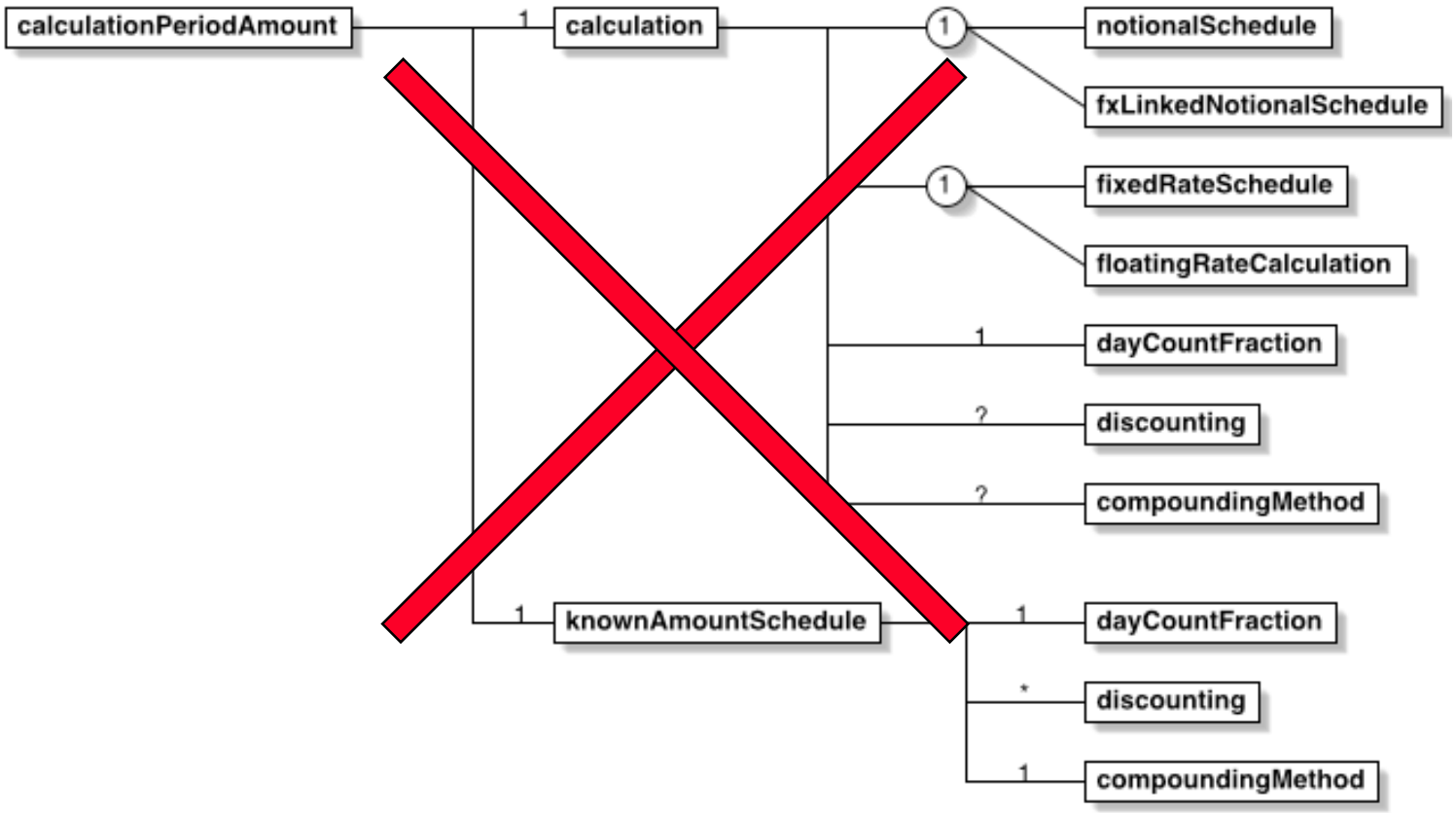
FpML Swap Structure



FpML Swap Structure



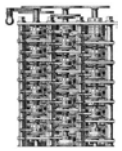
FpML calculationPeriodAmount component



FpML Swap Calculation Structure

Replacement for Payoff Component?

- Term sheets incorporate mathematical payoff formulae
  - E.g. $$\frac{1}{0.5 * n * (n-1)} \sum_{[X,Y]=1}^{0.5*n*(n-1)} PairwiseCorrelation_{[X,Y]}$$
- MathML is appropriate XML representation
  - Content ML captures semantics
  - Presentation ML for layout
  - Can be combined to replicate meaning and appearance of term sheets
  - *Example MathML...*



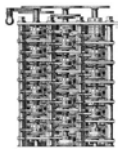
FpML Conclusions

Advantages

- FpML/MathML/XML because…
  - Emerging interbank standard for derivatives
  - Easy to incorporate MathML terms
  - Can combine FpML components into other instrument types
  - Utilise FpML applications & extensive XML toolset

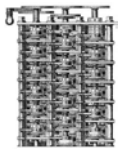
Disadvantages

- FpML limited because...
  - Instrument coverage falls far short
  - Continuous effort to anticipate structures
  - Lack of recursion & inadequate flexibility
  - MathML extension essential
  - Cannot use standard FpML applications without extension

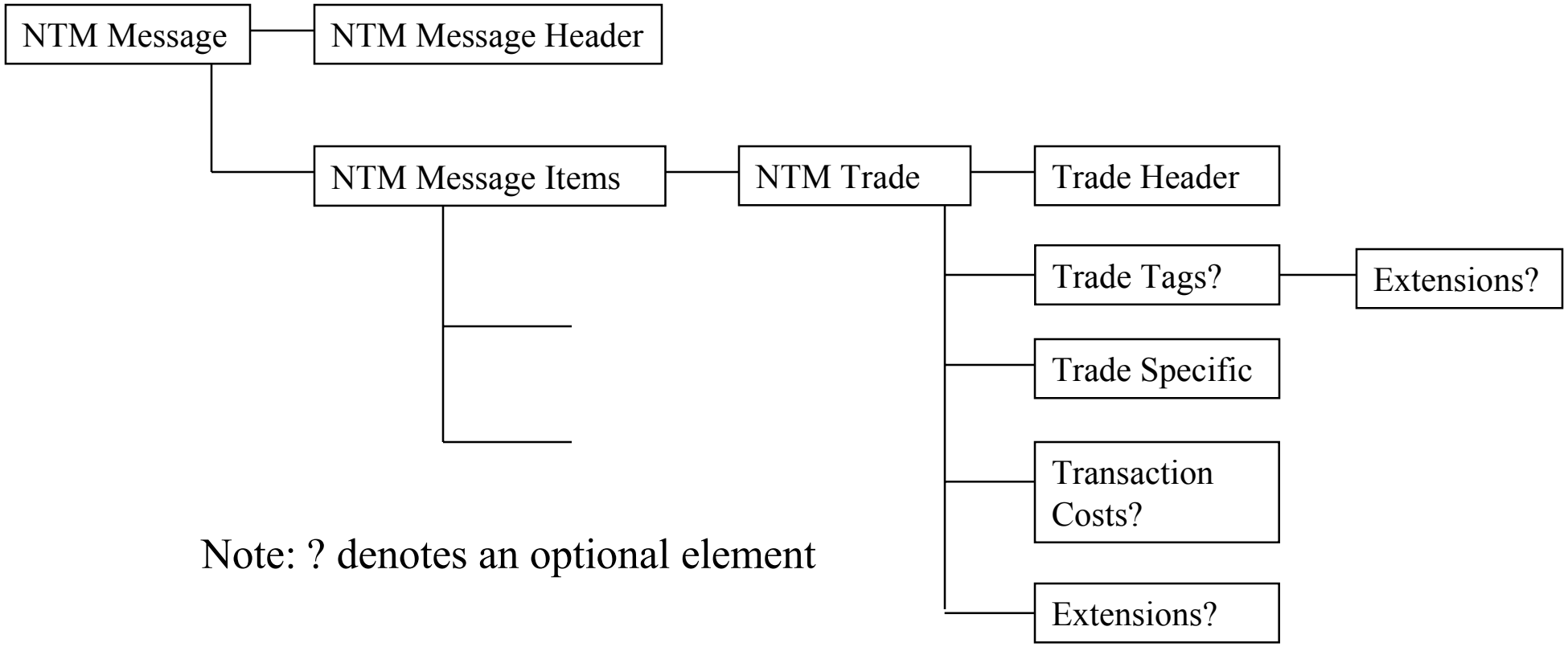


NTM (Network Trade Model)

- Freely available XML-based trade messaging protocol
  - Handles simple high volume structured products
  - Emphasis on describing trade state
  - SunGard states the goal as “capture 100% of the simple structured trading activity and at least 90% of the more complex interest rate derivative trades”
- Similar approach to FpML and restricted to a small set of standardized instruments with converters for the two systems available
- Cannot capture any detail of payoff functions, baskets contents or other contingencies with market events in a product



NTM DTD Architecture

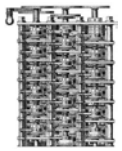


Note: ? denotes an optional element

Source: SunGard

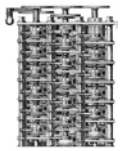
Valuation Systems

- Consideration of tools concentrating on representation for *pricing* rather than formal representation
- Pricing libraries of systems considered have a tendency to concentrate on *specific* asset classes
- Comparison to FpML
  - Proprietary to vendor
  - Extension of the form-based approach
  - More flexible in accepting non-standard contracts
- Use product scripting tools similar to those which exist within many banks



Reech Adep – Overview

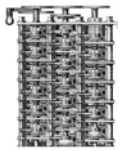
- Concentrates on product representation for *valuation* rather than *management*
  - Description boundary conditions and events using forms and scripts
  - Corporate clients buying commoditized products as intended client base
  - Strengths are mostly in Equity and FX derivatives
  - Not an STP (Straight Through Processing) solution
- Tools accessible to people who “may not be derivative literate”
- Application Service Provider
  - server farming with thin client interface



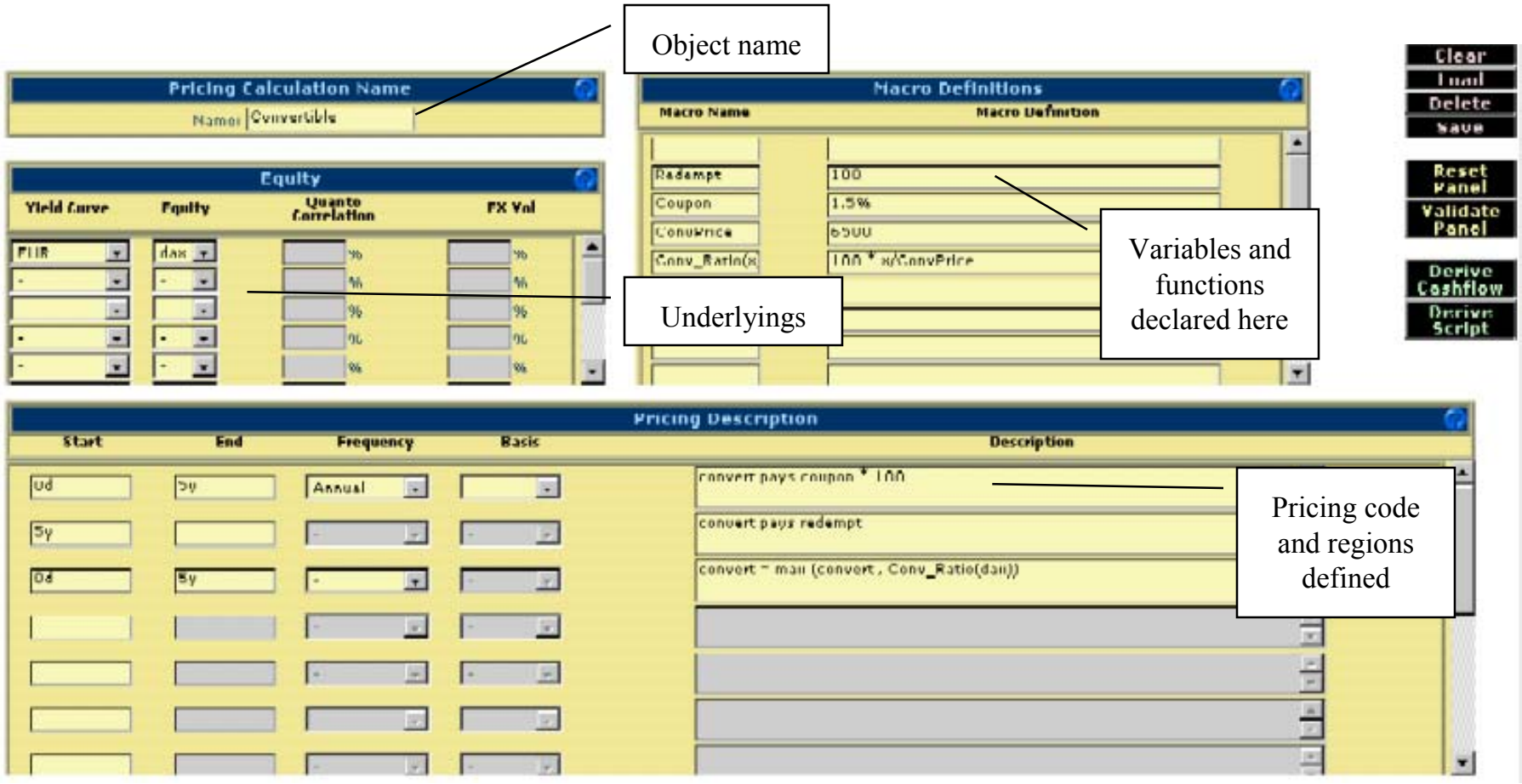
Reech Adep – Contract Description

- Traditional scripting language using ~35 keywords
  - BASIC-like syntax
  - Limited syntax checking and using only global variables (side effects!)

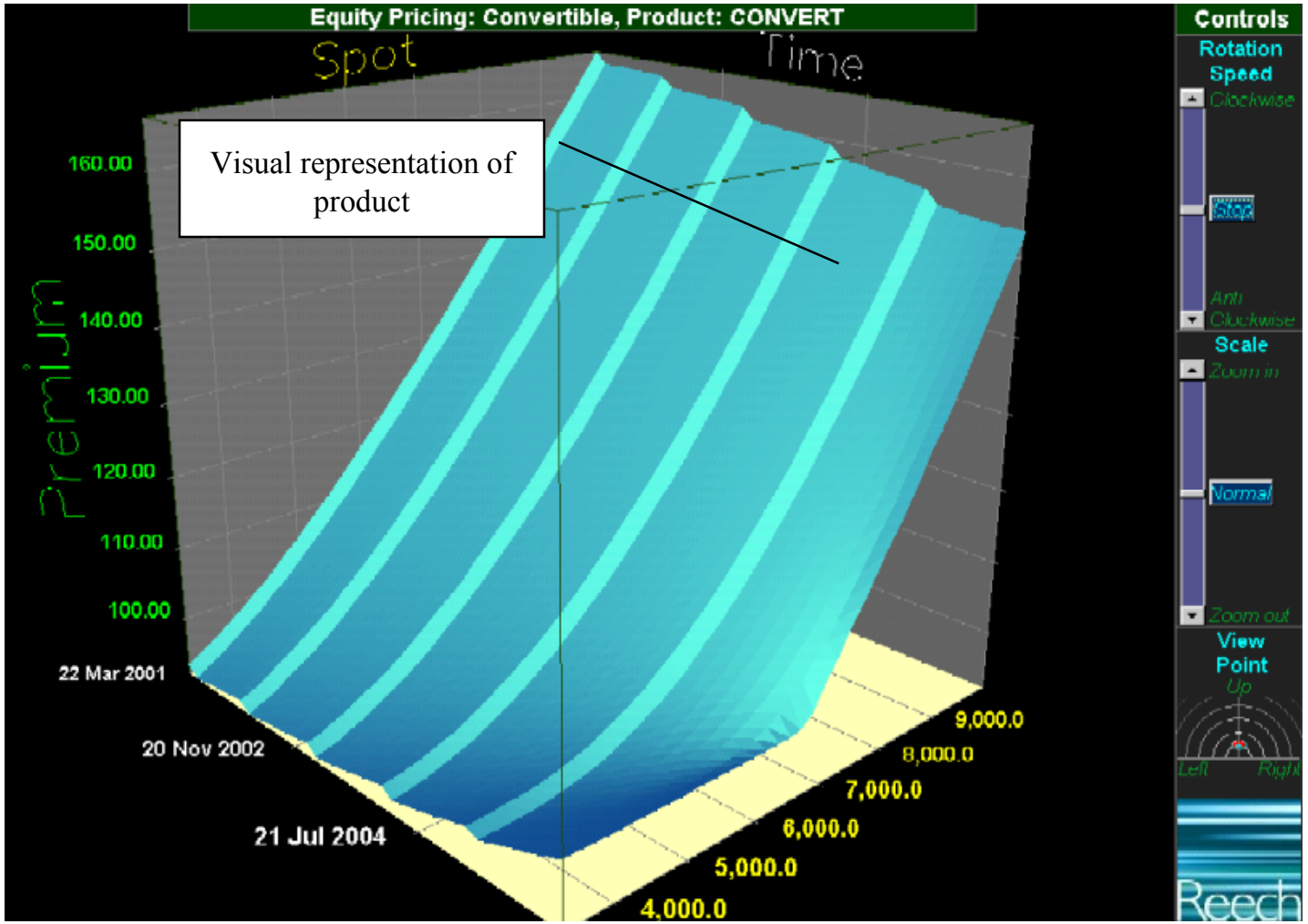
- Pricing procedure
  - Choose valuation / market model
  - Define asset pack and load a product template
  - Add macros and write actions for date ranges as required for intrinsic price
    - Global variables
  - Choose to price product using PDE/Lattice or Monte Carlo
    - various models available



Reech Adep – Pricing Example



Reech Adep – Pricing Example



Reech Adep – Pros and Cons

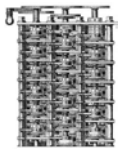
Advantages

- Packages a large body of pricing expertise
- Basic set of pricing models available which can be extended using API
  - handles relatively complex products out of box (e.g. mortgage backed securities) though Adep was extended specifically to handle this particular case
- Graphical user interface allows for multiple skill levels
  - well-developed graphing capability allows for visual debugging

Disadvantages

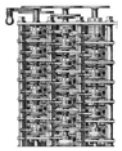
- Products that can be defined reflect pricing models and market assumptions
- Limited syntax and type checking

Not ideal for formal specification

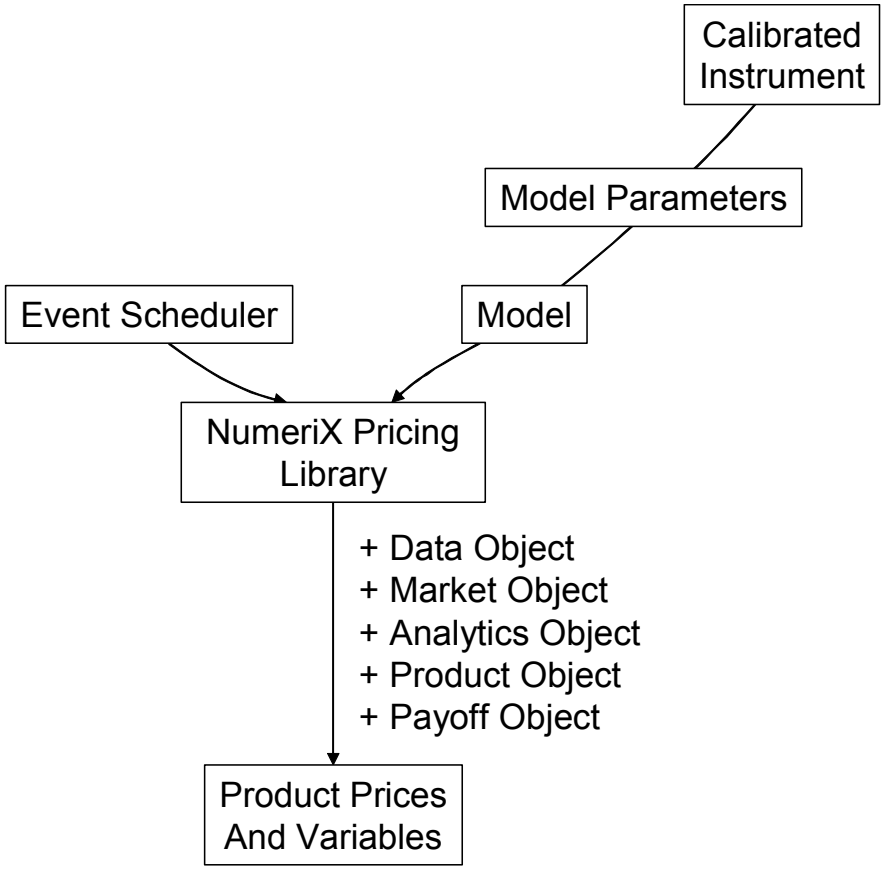


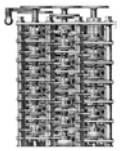
Numerix – Introduction

- Offers two product lines
  - NumeriX Engines
    - a mature solution
    - pricers written in a proprietary scripting language
    - interfacing with Excel or customers' existing systems
    - Components for: Credit Derivatives, Fixed Income, FX Derivatives and Cross-Asset Risk Management
  - Numerix Pricing Library
    - newly developed C++ library with pricing objects and callable solvers
    - will be integrated into NumeriX Engines
- Model setup does not necessarily require programming and C++ is not necessary in any of the cases



NumeriX Pricing Library





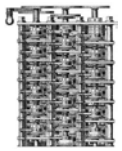
NumeriX Pros and Cons

Advantages

- Very strong pricing with good depth and breadth in the pricing libraries
- Good integration with Excel reduces need for retraining
- Cross-asset risk valuation available

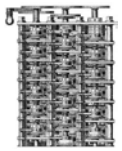
Disadvantages

- Capabilities in contract representation and manipulation limited
- Flat payoff function requirement limits combining of contracts



CygniFi

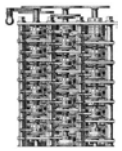
- Suite of products for derivatives trading and risk management originally developed by JP Morgan Chase where still in use
- FpML representations where possible
  - extended to allow representation of portfolios and market data
- Extensive pricing libraries available concentrated in fixed income and FX derivatives areas
- Product has been recently acquired by MB Risk Management – and others? – so the future of this product is not clear



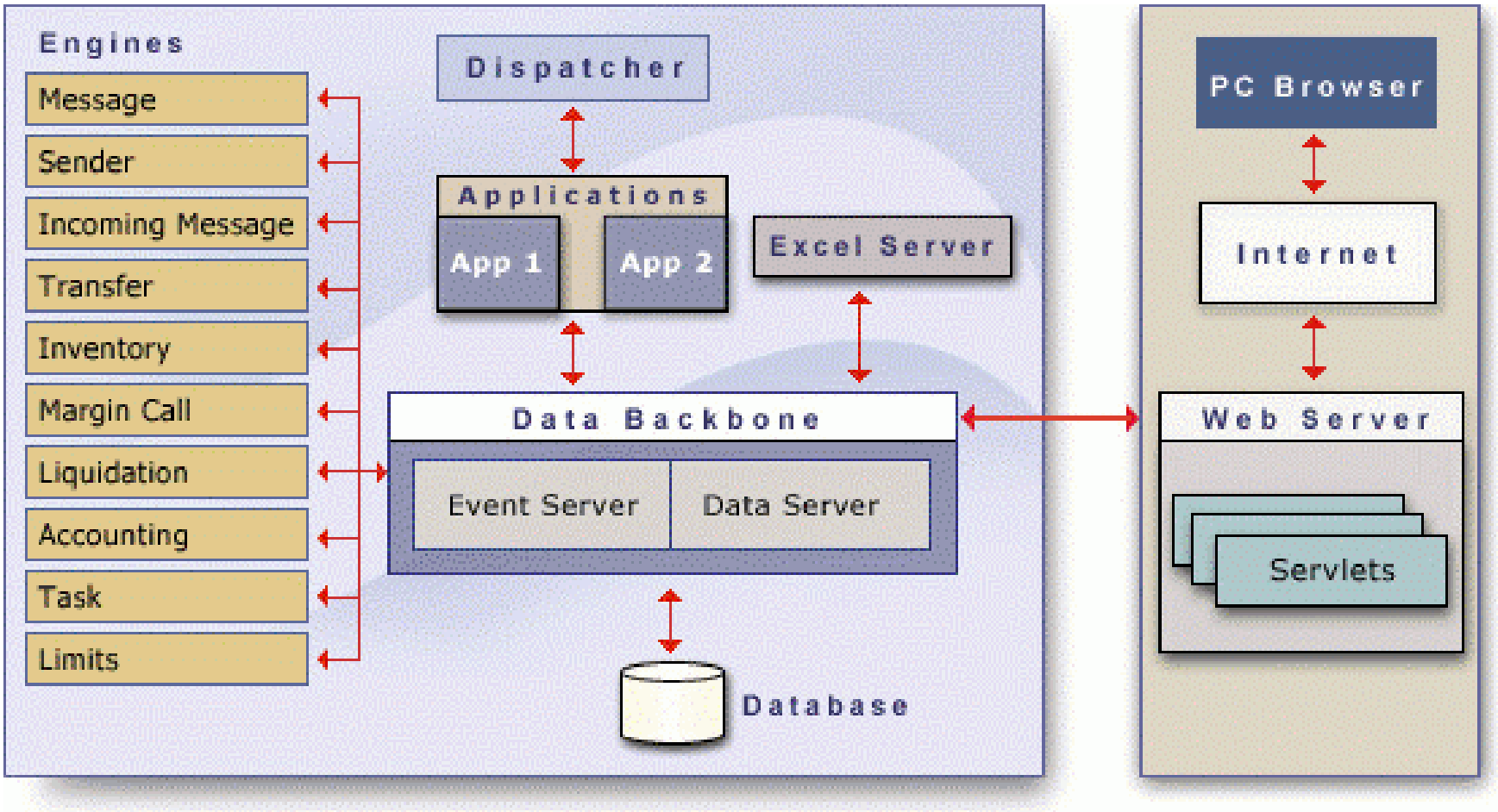
Calypso

- Developing an “Enterprise-wide STP solution”
  - ranges from trades to contract execution
  - Pricing appears to have come from the credit derivatives area in particular and is being extended to include FX and fixed income
- Product developed in Java to provide cross-platform compatibility
  - overhead of virtual machine may be a problem although JIT compiling is promising
  - integration of existing pricing libraries could be troublesome given limitations of Java
- Interface for Excel available this year

Important product to watch for the future!



Calypso Architecture

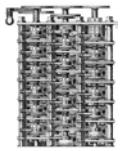


Source: http://www.calypso-tech.com

Formal Languages

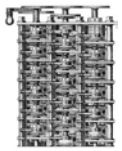
- We will consider Functional Programming languages rather than Imperative ones such as C++ because code written in the former has the features
  - more methods for composing functions (contract primitive combination)
  - verifiably correct (theorem proving!)
  - transformable – with multiple representations

- Examples include
  - O'Caml – base language for MLFi
  - XSLT – XML based language
  - F# and SML.NET – ML dialects produced by Microsoft
  - Haskell – a more pure functional language



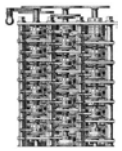
LexiFi – MLFi

- A ‘financial assembly language’
  - provides a minimal number *built-in* types including: **contract**, **observable**, **date**, **currency**
  - joined using a minimal number of *combinators* e.g.
    - **or** : *contract -> contract -> contract* – optionality
    - **acquire** : *observable -> contract -> contract* – define exercise conditions
    - **then** : *contract -> contract -> contract* – evolution of contracts
- Combinators are somewhat removed from commonly used and understood financial terminology but are chosen to be elementary
- Components can be built using O’Caml’s abstraction and modularization libraries for complex and commonly used instruments



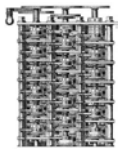
MLFi Pros and Cons

- ***Pros***
  - Allows precise expression of arbitrarily complicated contracts
  - Addresses questions of contract management
  - Opens up possibilities of automatic tool generation
- ***Cons***
  - Functional Programming may have high barrier to acceptability
  - Bank may become reliant on high-risk technologies and standards it does not control (e.g. MLFi, O'Caml)
  - May not provide an effective means of (e.g. interbank) communication if high-level usage is not agreed upon



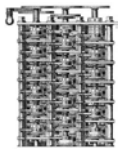
XSLT

- Verbose – it would not be desirable to use XSLT without some scripting language
- Not necessarily desirable to implement a MLFi style system in XSLT
  - Compiler availability questionable
  - Readability
- XSLT is better left to parse contracts rather than represent them



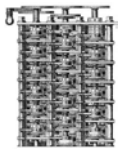
Automated Term Sheet Generation

- One of the reasons for using MathML-like syntax for equations is to simplify their conversion to equation presentation format
  - FP contract description syntax would *not* include the entire set of MathML commands
  - Commands *should* be transformable to MathML for layout in term sheets
- Certain comments in the code would be dumped directly into the termsheet
- However human intervention would still be required because of for example master agreements and limitations in machine capabilities!



Summary and Conclusions

- Require complete accurate representation of contracts
  - Multiple user groups
    - traders, counterparties, risk managers, back-office,…
  - Multiple uses
    - pricing, confirmation, risk measurement, settlement,…
- Term sheets inherently flawed
  - Arbitrary text
    - misinterpretations, specification errors
  - Readable only by highly trained staff not computers
- Existing data structures (Excel, XML) inadequate
  - No formal structure no standardisation
  - Incomplete semantics
    - data elements but not contract structure



Reviewed 3 classes of potential solutions:

- Product standards always in catch-up mode
  - FpML will not keep pace with OTC innovation
  - Aimed at wholesale inter-bank market
- Valuation systems are not generalizable
  - Tied to pricing models
    - no syntactic support of instruments outside scope
  - Proprietary standard not generally accepted if at all
- Formal languages too new
  - No proven commercial systems or financial libraries
  - Have to train existing staff in new methods

Still a long way to go!