Chapter 5

EM For Integration and Virtual Collaboration In the Financial Enterprise

5.0 Overview

This chapter[1] establishes a framework for the application of the Empirical Modelling technology in the financial enterprise. Two applications in the area of software engineering and software system development are considered: ***software integration*** and ***virtual collaboration***. Section 5.1 considers the problem of software integration in the financial enterprise. Section 5.1.1 motivates financial software integration with specific reference to a challenging exercise involving the integration of ERP and e-commerce applications. Section 5.1.2 discusses issues relating to financial software integration and overviews the current approaches to tackling these issues. Section 5.1.3 frames the challenges for financial software integration and proposes new principles based on the EM approach to address them. A Situated Integration Model is proposed to meet the software system development agenda for financial software integration. Section 5.2 considers virtual collaboration in the financial enterprise. Section 5.2.1 overviews various forms of collaboration. The importance of taking human information behaviour into account in virtual collaboration is discussed in section 5.2.2. Section 5.2.3 considers the challenges to virtual collaboration. Section 5.2.4 proposes new principles for virtual collaboration based on the EM approach and illustrates these with reference to the case study of online trading. Section 5.3 concludes with the future prospects of EM Technology in the financial enterprise and speculates on the theme of building a web-based environment for corporate intelligent networks.

[1] This chapter is an expanded version of the combination of two joint papers by the author and Meurig Beynon [BM99, BM00].

5.1 Software Integration In The Financial Enterprise

This section addresses an important problem facing almost all financial institutions re-engineering their processes in the face of global competition pressures: the integration of existing technologies used in routine daily activities and in intelligent decision making.

Whereas there is a massive literature on the development of standalone software system applications solving problems in the real world domain, relatively few references can be found on the basic principles and practical applications of the integration of different standalone software systems. This represents a wide and deep gap in software system development and software engineering studies. ***Software integration*** embraces both ***technical*** and ***social*** aspects. The technical aspects relate to data and operations on data. The social aspects relate to the mode of interaction of human agents accessing the data and the context and environment within which this interaction takes place. Key issues for financial software integration to be discussed in this chapter include: a clear definition of the term "*integration*" and its scope and use in software system development; basic issues invoked in the integration; and techniques to solve the problem of integration.

Financial software integration is a situated context dependent activity. Considering a specific practical exercise in software system integration is only of limited benefit in identifying a methodology for software integration. Empirical Modelling technology proposes a framework for financial software system integration based on basic principles and techniques that inform the requirements engineering for software system integration. It promises to deliver two models: a situated integration model (SIM) to be used in the requirement engineering phase of the software integration process that combines conventional approaches to software analysis with the EM approach to software development and program comprehension; and a model of an integrated system accounting for views and agents within the system. The SIM helps in exploring possible modes of interaction, developing algorithms for integration, analysing distributed and shared access to data, combining visual interfaces, synchronizing the data flow, and providing a unified functionality.

5.1.1 The need for integration

Understanding, analyzing, and constructing experimental models of real life financial systems is a wide-ranging task that requires an integration of different technologies and enabling tools and calls for a bridging of the gap between theory and application as well as research and development in this area. Business process modelling, intelligent state and agent-oriented modelling, data warehousing and data quality assessment tools, financial analysis tools, and client server technologies should tie up coherently to enhance knowledge acquisition in a global financial market. Players in the global financial marketplace, such as investors, rating agencies, financial service providers, analysts and consultants, are faced with a massive amount of explicit and implicit market information characterized by a high level of dependency and interrelationship. An automated environment which, as far as possible, depicts and captures all aspects of the real-life financial system, is needed to adapt to state changes in global financial markets. Such an environment should provide a flexible human computer interface backed with a high level of interaction, visualization and reporting capability, with minimal overhead coding requirements.

Financial institutions are amongst the largest investors in computer technology and therefore appreciate the importance and significance of such technologies for their growth and survival. Financial institutions have often exploited technologies to create innovative products and services, capture market niches, and better serve the customer. The use of IT in the business and financial sector has evolved from the simple electronic data storage and limited computational capability of large business and financial databases to electronic data analysis and interchange. In the process, new forms of electronic data storage have been introduced, so that relational database management systems have been complemented by object-oriented databases and hyper-based storage. "Turning information into knowledge" is a major corporate challenge and the rate at which organizations learn and accumulate knowledge may become the only sustainable competitive advantage. With the growing complexity and competition in the global markets, the effective use of scarce resources and new technologies is of strategic importance. Enabling tools and technologies, such as business process modelling, intelligent analysis, data warehousing, and web technology should integrate coherently to establish corporate intelligence networks for intelligent information gathering, dissemination, and decision making. This intelligence network is critical for enabling global knowledge and information consolidation and distributing heterogeneous data relating to local markets (cf. Figure 5.1).

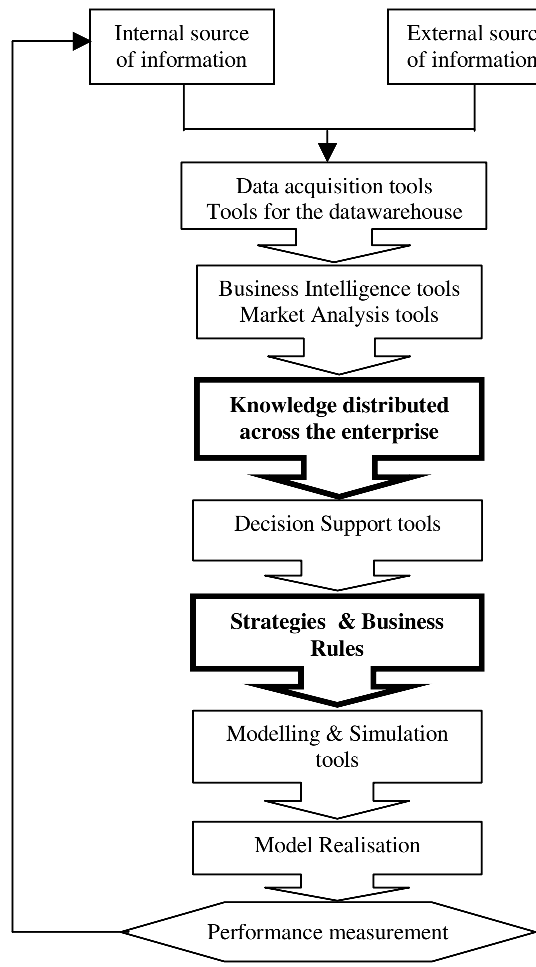

Figure 5.1 The integrated application chain

The urgent need to research various issues related to financial software integration is highlighted by considering the case study of the integration of: i. e-commerce applications with ii. enterprise resource planning[2] (ERP) applications.

[2] Enterprise resource planning (ERP) is an umbrella term for all tools and technologies used to handle the internal operations of a firm and to automate its business processes (ERP applications include controlling payroll, inventory, purchases, finance, personnel operations, etc.). The term *back office* refers to the IT centre where all enterprise resource planning applications are handled. ERP software applications used to run on mainframes, however, the advent of the year 2000 problem, and the introduction of the euro currency has forced many business firms and financial institutions to upgrade the tools and technologies used in their back-offices and to re-engineer their internal processes. Object Oriented technology is adopted in the design of many of today's ERP applications.

i. The emergence of the internet has challenged traditional business models through its ability to offer direct routes to market, to reduce barriers to entry and to increase the efficiency of trade activity [TW99]. Electronic commerce has been experiencing explosive growth[3]. Firms started their electronic commerce activity by first establishing a web presence, then by promoting some of their products and services online. These initiatives were not so expensive, they generated adequate profits, and increased the firm's interest in e-commerce activity. Today, electronic shops[4] are becoming more advanced, their design is more complicated, and their rewards and costs are higher. E-commerce applications have been developed with a view to establishing customised online stores with various design and catalogue structures. The workflow of transactions between the suppliers and buyers is becoming more efficient and less error prone when handled electronically. In a competitive global market, shops and trading firms selling their products and services online are facing a major question: how useful, profitable, intelligent and attractive is an online trading store?

ii. After a long period of sustained growth, the market for enterprise resource planning applications reached a saturation level and started its downturn. This is attributed to many factors including the diversion of funding to fix the year 2000 problem, the saturation of the ERP market, the high cost of acquiring and maintaining ERP applications, and the difficulty in introducing changes to these applications due to the inflexibility of their design and structure.To restore their viability and to tap into new

---

[3] Results of a recent survey from Visa International confirm that commercial electronic commerce is set to grow dramatically in Europe over the next five years, at a rate of about 30 times the growth of GDP in most European countries. Another e-commerce survey, conducted by PricewaterhouseCoopers in June 1998, predicts that within the next three years the major growth areas are extranets for business partners, company-wide data warehouses, customer service via the internet, web catalogues, and web-based transactions.

[4] A typical virtual shop model includes: (1) The *contact information* that provides address details and shopping support contacts. It also offers communication via multiple channels including e-mail, voice and video communication; (2) The *electronic catalogue* that contains information about all the products and services offered by the virtual shop. Electronic catalogues can have different presentation formats, customised to different types of products, and satisfying the customer preferences for viewing these catalogues; (3) The *surfer that* allows customers to intelligently surf the electronic catalogue, recording their sales preferences in their own *profile; (4) Options for c*ustomers to place buy *orders* online and to *enquire* about the history of their order transactions; (5) *online contact with suppliers and partners*, facilitating inventory control, replenishment, and management; (6) *Online handling of Shipment* orders to assure an adequate stock level of all items and products in the electronic catalogue.

markets, additional features and capabilities are being added to ERP systems, extending their role beyond integrating and automating business processes. These new features include customer relationship management[5] (CRM), enterprise Web portals[6], supply-chain collaboration[7], business intelligence[8], data warehousing, Web-enabling ERP[9], and other added value features [Twe99].

A successful integration between e-commerce and enterprise resource planning (ERP) applications is a crucial factor in effective e-commerce. However, despite the growth and rising demand for e-commerce activity, the corresponding tools and applications are still residing at the front end, and remain somewhat divorced from the back-end enterprise resource planning applications that handle payroll, project planning, business processes,

---

[5] Customer relationship management (previously introduced in chapter 2) is referred to as the ability to capture customers and to satisfy all their needs and requirements with minimum cost and high efficiency. This relies on many tools and technologies that can capture all the relevant information about customers and their needs via multiple channels (e.g. telephone, fax, internet). This information is then stored in databases, and is analysed using data-mining and business intelligence tools to detect profitable customers and to turn the normal call center into an intelligent one.

[6] An ERP portal provides users with a single home screen (web-based) from which they can access all the ERP applications and data sources they need to do their job. The ultimate goal of enterprise web portals is to provide companies with a more efficient channel for delivering self-service applications [Eas99].

[7] In commerce and industry a great deal of effort is made to establish an end-to-end electronic supply chain. This will potentially help in maintaining minimal stock while satisfying customers needs, and in providing low-cost efficient communication with partners. Early attempts at supply-chain collaboration using internet-based trading systems failed due to the difficulty of integration with existing enterprise resource planning applications, the high cost incurred, the lack of security, and the high risk of system failure [Gur99]. Supply-chain collaboration has evolved with more standardised, flexible and efficient electronic interchange systems and technologies.

[8] Business intelligence applications simulate human expertise and reasoning, and operate on massive amounts of data. Within this category we can identify OLAP (Online Analytical Processing), data mining, reporting and analytical applications.

[9] Web-enabling ERP applications means providing a common browser interface to allow employees, customers and partner access to personnel detail, inventory information, or other information in internal systems. Web-enabled ERP is the extension of existing traditional ERP applications to the internet. Web-enabled ERP applications use a thin client architecture rather than a client/server architecture [Eas99]. The main driver behind web-enabling ERP applications is the maintenance of a low cost of ownership in the wide deployment of these applications. Full featured desktop solutions, often called fat clients, are expensive [Hum98]. The total cost of ownership, a common measure of return on investment, brings into a single view all related costs of a technology including capital, administration, technical support, and end-user operations. However, despite the great interest in web-enabling ERP applications, there are still many challenges. These challenges are faced by the current approaches to integration overviewed in the following section.

accounting, and intelligent processing of data. Many attempts have been undertaken to integrate e-commerce and ERP applications. The challenges facing this integration vary in type and scale depending on the technology used at the back-office. Integration with legacy systems, as compared to the integration with modern ERP systems via middleware, poses many challenges.

The complexity of the integration problem presents a challenge to software engineering in respect of system representation and analysis. There is a need for new modelling techniques and principles that can cope with the complexity of the interaction between programmable components and human agents. Viewing the internal ERP system from a web front-end entails many complications that pose fundamental challenges to currently available ERP technologies.

E-commerce applications need to interact with a number of ERP applications (such as sales, order management, payment, fulfillment and customer administration). Integrating a virtual shop model with a suite of ERP applications including purchase, payroll, accounting, and inventory control applications is depicted in the figure below.

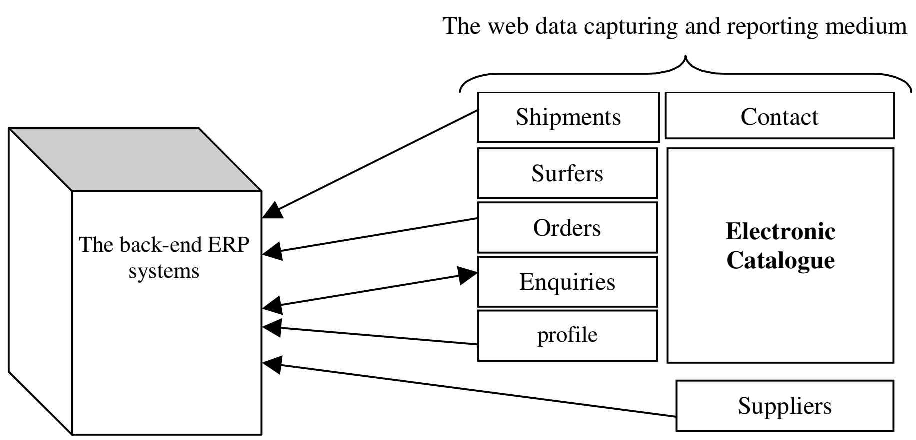

Figure 5.2 Integration of e-commerce (Virtual Shop Model) and ERP applications

Where this integration is not established, the data captured from the web front-end fill-in forms is mediated manually or semi-manually to the ERP purchase application. The response to online customer request for their transaction status or any other information is served via a separate communication medium such as the fax, post or phone, with a considerable delay in the execution process. Moreover, unexpected purchase orders for some products may be delayed if the electronic supply and buy chain is not closed.

The integration of e-commerce and ERP applications provides a straight through processing of customer orders and enquiries and enables stores to hold the necessary level of inventory.

In this context, the integration of e-commerce and ERP applications is viewed as encompassing: web enabling ERP applications, extending and improving the online customer service and the online reporting, and the straight through processing of customer orders and enquiries.

This integration has been approached from many different perspectives. The challenges faced vary in type and scale depending on the technology used at the back-office.

- *The integration with legacy systems*: Connecting web front-end to old legacy systems is very difficult, because these systems make no distinction between the stored data, the structure of this data, and the application. It is hardly possible to modify the data without using the legacy application, which is often undocumented and requires a lot of time and expertise to understand [Han99]. An easy, but unsatisfactory, solution adopted by many firms engaged in the integration of legacy ERP applications with e-commerce front-end applications is to take the information captured online and retype it into the old systems. Another problem facing the integration of e-commerce applications with legacy ERP systems is the need to synchronise the old batch processing of data with the online flow of data captured via the internet browser.
- *The integration with modern systems:* Modern ERP systems consist of two logically distinct elements: a database (which holds the information) and an application (which does the processing). Data stored in databases is then easily manipulated (extracted, filtered, sorted, etc.) using a query language, without the need to understand the application [Han99]. Recent attempts at integration involve linking modern ERP systems with the web front-end via middleware[10].
- *Integration solutions offered by ERP vendors:* Many ERP software vendors are web-enabling their ERP applications, allowing access to these applications from a desktop or a browser interface. However, this approach lacks flexibility and customisation and no one can predict to what extent the problem of shared access to information is properly resolved.

[10] Middle-ware is software that sits between the data sources and applications.

In all the above-mentioned attempts at integration, major common problems are faced. These are mainly due to conflicts arising from shared access to data, poor data quality[11] and rigid architecture and design[12].

5.1.2 Systems Integration Perspectives and Issues

In its broad sense, ***integration*** refers to the coherent merging of two entities, having different behaviours and attributes, to obtain a unified entity that can realise the behaviours of its components and whose attributes are derived from but are not necessarily the same as those of its components. Integration might be necessary to accommodate a new style of relationship between different entities, or it might be optional with the aim of enhancing performance and gaining value-added advantages.

The term integration is used in many different contexts; it can refer to economic integration, horizontal or vertical integration of companies, software integration, system integration, etc.. We concentrate here on the integration of IT systems, focusing mainly on software integration over a distributed hardware configuration.

The first precedents for IT integration are to be found in the databases of the late 1960s and early 70s. At that time, databases typically displaced suites of application programs based on different file record formats and assumptions about physical storage [KS86]. They also reduced the need to commission new programs or re-engineer existing programs to achieve new functionality. The key principles that emerged in this process were *logical data independence* and *physical data independence*. Arguably it was also established, whatever limitations of relational databases have subsequently become apparent, that relational theory and the analysis of functional dependencies in data are an essential aspect of maintaining successful data independence.

---

[11] These problems arise from the inaccurate recording of information in the internal systems and from much duplicate data [Han99]. Connecting web front-end to ERP applications might violate the data integrity and reduce its reliability if data quality issues are not carefully addressed.

[12] Despite their popularity, ERP applications are criticised for their monolithic structure and inflexibility, the difficulty of combining them with third party bespoke components, the difficulty of component-wise upgrade to new releases, and the high cost of their customisation, maintenance, and training [SSD98].

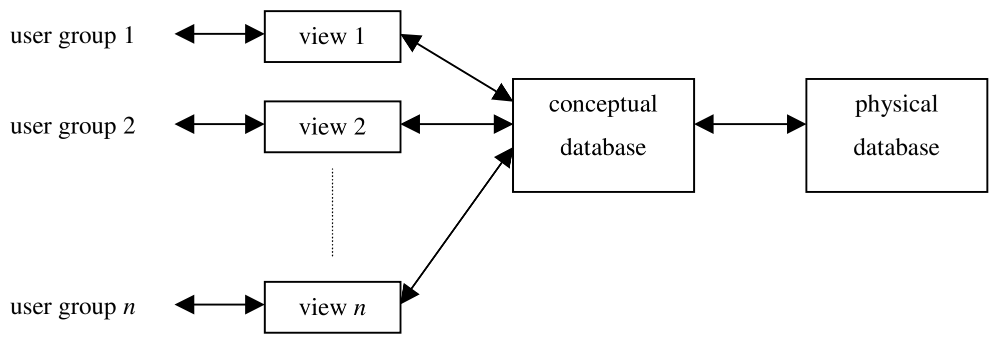

Figure 5.3 Logical and physical data independence

It is now clearer than it was at the time that relational theory supplies a database solution well-suited to a particular kind of business process model. The success of relational databases in areas such as banking depended on the highly routine nature of the transaction processing and the uniformity of the data representation demands. Subsequent developments in computer applications have exposed the need for greater flexibility in database technology than current commercial relational products have been able to deliver. The problems of integration in contemporary applications such as financial systems are further complicated in a variety of ways:

- whereas in the 70s data input was typically manual, and there were no critical real-time issues to be considered, automatic data acquisition via programs or sensors is now common;
- information is now accessed and processed for interpretation in far more complex ways, e.g. through graphical user interfaces, Business Intelligence tools and report generators;
- there is now a demand for large-scale integration of what were formerly quite distinct divisions of business activity, as in the trend to concurrent engineering, datawarehousing and comprehensive business process models.

Where state-of-the-art financial systems are concerned, these problems are compounded by the factors discussed above that promote volatility and instability of the business process model.

Classical approaches to software integration are typically based on combining two paradigms: the use of relational methods to construct integrated data models, and the use of object-oriented methods to describe the operational processes surrounding business data at many levels of abstraction. Both relational and object-oriented modelling methods can be seen as addressing data and agency to some degree.

Functional dependencies in a relational model express the way in which a change to one item of data has to propagate change to other items in a way that is conceptually indivisible if data integrity is to be respected. In the relational theory of database design, they also supply the framework for organizing data to meet the needs of different users. The characteristic mathematical abstractions of relational theory do not capture the distinctions associated with different kinds of agency on the part of the user, however. A relational algebra expression can be used to express how one relational table is derived from others, but this may be used to evaluate a one-off query, to construct a virtual table or view, or (e.g. if associated with a spreadsheet interface) be subject to a process of continuous update in response to independent interactions with its operands. Where the human agent is in full control - as in traditional database applications - distinctions of this nature can be effectively managed without reference to explicit models of agency. Where the database is interfaced directly to computer programs, or to electronic devices with an external interface, the need for an explicit way of expressing and analyzing agency becomes evident. For instance [CW98], two COTS programs that access the same database table may operate quite effectively in isolation, but fail when integrated because one locks the entire table in order to access a single tuple.

Object-oriented models focus upon modelling collections of data together with the fundamental operations that can be applied to them. In this respect, they are well-suited to representing the components of distributed systems, as is appropriate in a typical context of virtual integration. What is lacking is an effective way of dealing with the complications that arise from concurrent interaction between objects. This problem has both *accidental* and *essential* aspects in the sense of F P Brooks (1995). It may be that appropriate re-engineering and the use of frameworks such as CORBA and DCOM can resolve the accidental complexities introduced when processing over several machines in a network, using different programming languages, or running on different platforms. To apply these techniques effectively, it is still necessary to resolve paradigm differences, as when trying to convert a standalone legacy system designed without object-orientation into a modern three-tier architecture. Even supposing that this can lead to a homogeneous collection of objects communicating seamlessly, essential problems remain. It is necessary to account for active objects and autonomous agents and for the perspective that subject-oriented programming provides [HO93]. There are challenging and well-established problems concerned with the operational semantics of concurrent object-oriented systems. A strong body of evidence from relational database theory also argues against unsupported object-orientation as a solution to data and application integration [COM94].

Issues concerned with agency are amongst the most difficult to represent and the most subtle to analyze and resolve. Operating systems provide the setting in which such issues have generally been encountered hitherto. It is hard to prescribe automatic solutions for the common problems of contention, synchronization and conflict that can arise when many applications are integrated. Experimental activity and insight specific to the particular situation typically rule out full automation. This is acknowledged in the design of GENIO, a software tool for data integration that has been described as a "databroker" [Por99]. GENIO gives the user the means to control the scheduling and propagation of change between data representations either through direct personal intervention or by supplying parameters for automated data conversion agents.

5.1.3 Framing the challenges of software integration

The most important issue in software integration is the plethora of ways in which data is accessed and processed. This can be interpreted as a need for better models of data and agency. The focus is no longer on abstract data alone, but on the state-changing activities that surround that data, to include the protocols and interfaces of all the agents that operate upon it. In this context, an *agent* can refer to a computer program or procedure, a human agent, or an electronic device that mediates between the internal representation and the external world. With this interpretation, agency is manifest at many levels of abstraction, in low-level data exchange, in internal business interactions, and in the external business environment. In general, the problems of integration cannot be resolved without taking account of the multiple views imposed upon data through different types of agency. Only in quite exceptional circumstances, when there is an unusual degree of consistency in the ways that data is addressed and modified by agents, is integration of data representations sufficient. Several key issues have to be addressed for successful IT integration:

- dependency and the indivisible propagation of state change;
- the association of data with operations upon that data;
- the modes of agency by which state-changing activities are mediated and synchronized.

Successful integration gives users and programmers concurrent access to many software applications in a distributed environment (Figure 5.4). It must respect the integrity of data in relation to user views and the external environment. It involves combining interfaces both at the user level and at the machine level, where it provides shared access to raw data stored in system files, taking account of priority and effect. Whilst each of these issues has been addressed individually by current approaches to IT integration based on a relational, object-oriented or data-broker models, there is a need to combine the qualities of all three.

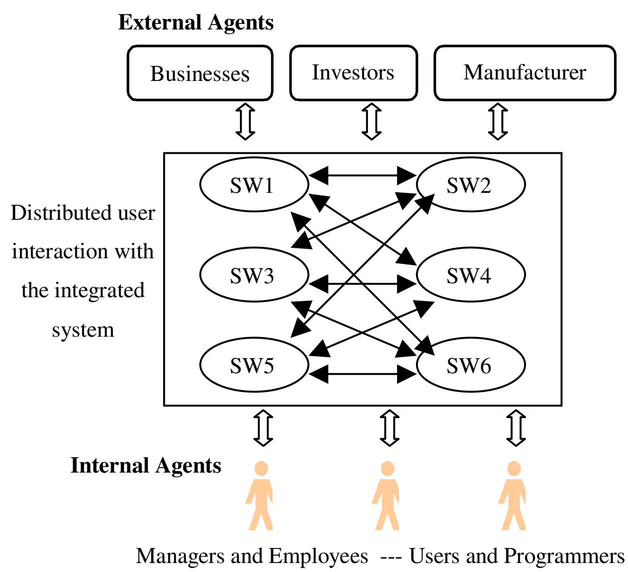

Figure 5.4 An integrated agent oriented system

5.1.4 The Situated Integration Model (SIM)

Software integration in general combines two kinds of activity: integrating existing separate software products and applications, and constructing new multiple-purpose software components. In this exercise in software re-engineering, the principal agenda is:

- interface design and interaction;
- shared access to data;
- synchronization of extraction, transformation, and loading of data;
- creation of a coherent and unified suite of functions.

Current proposals for software integration involve creating a metadata repository comprising profiles of each of the different applications to be integrated. Each profile is compiled from existing documentation and from the results of manual or automated code analysis. When combined with suitably engineered Common Information Models, the metadata repository supplies the resources from which the integrated system is to be developed.

The creation of a metadata repository is realistic only in certain contexts. It ideally requires the source code, if not the requirements and design documentation, for each component application. The analysis activity is an exercise in program comprehension that can be very challenging if it involves a mix of programming paradigms, or ill-documented legacy code. At best, the repository provides such documentation as is associated with requirements capture and specification in a modern software development method. This provides a static view of the system supported with static workflow models. Even where this information is sufficient to specify the behaviour of individual software components in detail, this does not address the agenda for software integration identified above. Successful integration requires crafting of the corporate behaviour of component applications, to include the specification of interfaces, strategies for shared data access and synchronised data processing, and the design of a coherent functionality.

EM principles for software integration are well-oriented towards the key issues of data dependency discussed in the previous section. Definitive scripts deal explicitly with dependency and indivisible propagation of change. The definitions in such scripts can be grouped in many ways, as their order is not important. By collating the state observables of an EM agent, as specified in LSD, an object-like abstraction is obtained. The permissible operations on such an object in general include redefinition of its state observables through the direct action of other agents. The operational effect of concurrent agency can be empirically established by simulating the execution of agent protocols, for example by using the EDEN interpreter. In this way, two or more modellers can play the roles of agents within the system independently.

Definitive scripts provide a powerful means of data integration that can be used in particular to express the way in which low-level redefinition can entail high-level change. Consider for instance how the interest rate can change when a balance crosses a threshold. Data conversion agents that are empirically tuned to particular patterns of synchronization can serve as databrokers. In EM terms, what has been described in Figure 5.1 as the conversion of data to knowledge is merely one aspect of pervasive mechanisms and processes that mediate between the viewpoints of one agent and other.

In EM, constructing an ISM addresses requirements understanding for software development in a way that circumvents the limitations of normal documentation. An ISM represents knowledge in an implicit and experiential manner. The modeller can develop, access and explore understanding through interaction with the ISM, and share this insight by presenting the ISM to another modeller. ISMs constructed using EM principles are so general as to encompass traditional engineering or scientific artefacts that are devised to capture empirical insights. Experimentation with legacy components of a software system can be used to develop ISMs in a similar way.

In applying EM to software integration, the key idea is to understand each software application in agent-oriented terms with reference to the particular observables that mediate its interaction with other agents in the system.

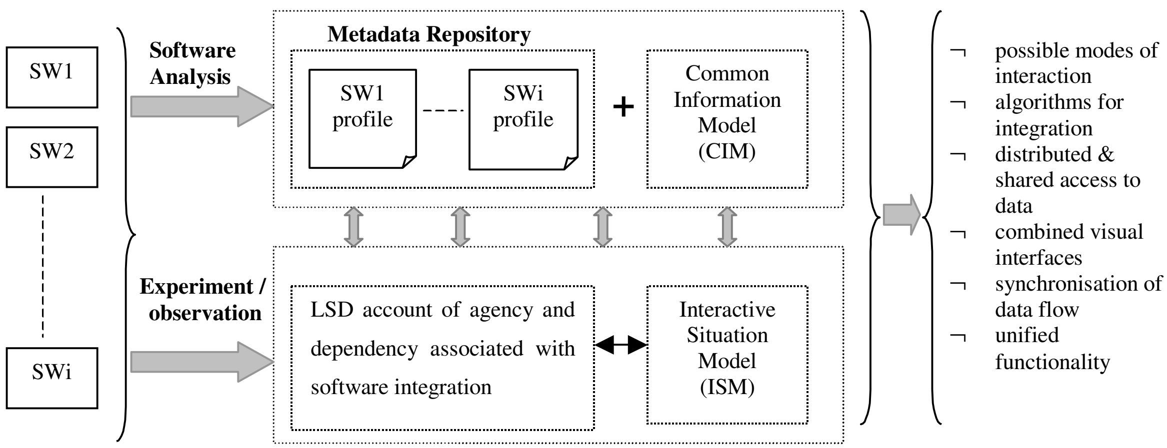

Figure 5.5 The use of EM to complement the conventional approach to software integration

This analysis is not simply concerned with abstract inputs and outputs, but with the way in which interaction is embodied at the interfaces to other applications. This embodiment of inputs and outputs is metaphorically represented by an ISM and its associated LSD account, as developed in parallel. Such ISMs enable an experimental study of the modes of interaction between both existing software applications and those yet to be developed (Figure 5.5). They can also help to address issues of scalability and customization.

The development of an ISM for software integration, to be referred to as Situated Integration Model (SIM), can draw upon ideas and techniques introduced in previous research. These include:

- the construction of ISMs based on animating conventional static artefacts (such as object models and statecharts), both to introduce models to the data repository and to refine and to exploit them [BCSW99];

- collaborative interaction of potential users of the integrated systems (such as the internal and external agents) through networked ISMs in a distributed environment [BS99];
- the intervention of the modeller in the role of an all-powerful agent (e.g. to shape the synchronization of interaction, to resolve conflicts between viewpoints and to compensate for incomplete rules for data brokering) [Sun99].

The EM approach also promises to deliver a model of an integrated system. This model is typically neither static nor comprehensive in character, but comprises a loose association of ISMs constructed from the viewpoints of different agents within the system, each reflecting different observables, dependencies and types of agency. Integration is achieved through a dynamic empirical process of negotiation between these viewpoints. This process in general entails compromise, and may require intelligent intervention by a human agent acting in the role of an arbitrator or broker.

5.2 Virtual Collaboration In The Financial Enterprise

5.2.1 Forms of virtual collaboration

The term *virtual collaboration* refers to collaboration via an electronic medium. The following paragraphs overview different forms of virtual collaboration, and discuss their uses and limitations [BM00].

a) Project/group work collaboration

In current group-ware, documents and document-related processes define the logical context for collaboration [Mar98]. Documents are where most corporate knowledge is captured, hence the importance of document-centric collaboration that is more structured. When teams collaborate on a project, the results of the collaboration are typically captured in documents that need to be maintained and managed. Technologies for collaboration then centre around tools to create, share, and distribute documents. The internet is the most common platform for document-centric collaboration. However, as observed by Ciancarini et al (1999), the web in its current state does not provide support for document-centric applications like group-ware or workflow that require sophisticated agent coordination. In this context, the term *agent* refers both to entities which can act autonomously and can receive/send messages according to some pre-defined protocol, and to human agents with assigned roles in the group work activity or project.

b) Collaborative learning

Early attempts at online collaborative education were motivated by the desire to explore technical advances in networking and communication rather than by well-defined educational goals. Experience has shown that generic network tools, such as e-mail, computer conferencing, and newsgroups, are weak in supporting collaborative learning [Har99]. This is attributed to several factors. These include: the lack of standardized ways to organize educational material; the overhead work to manage and monitor students' performance; and the lack of models to support learning strategies that involve knowledge building and sharing. Current online educational tools support collaborative learning and course management to a greater extent. These tools are mainly web-based, and include personal workspaces for students, course structuring, grade management, file management, and system management utilities. The Virtual-U web based learning environment (see www.vu.vlei.com) is one example of online learning tools. In addition to online educational tools, research and assessment tools have also been developed to study and analyse the behaviour and teaching/learning processes. Today the main challenge facing the development of web based collaborative learning environments is to support interaction with web pages that is richer than mere front-end access to static information.

c) E- business

The telecommunications revolution and the growth of internet activity have challenged traditional business models by offering direct routes to market, reducing barriers to entry and increasing the efficiency of trade activity [TW99]. E-business is an umbrella term for e-commerce, supply-chain collaboration, online trading, and business to business online communication. Business today is converging on the internet. This creates a great opportunity for organizations to communicate and share data over the web with customers, partners and suppliers. However, the growth and profitability of e-business activity is inhibited by many technical and social problems, including:

- technology integration (the integration of the back office and front office systems)
- the adoption of a common e-business model
- security

- the introduction of new national and international legislation that protects the rights of all business parties in executing cross-border transactions
- the deployment of a low cost efficient solution for true flexible business collaboration
- the high risks of system failure.

5.2.2 Human information behaviour and information horizons

Following D. H. Sonnenwald (1999), the term *human information behaviour* is used to refer to collaboration amongst individuals engaging with information resources in information exploration, seeking, filtering, use and communication. Sonnenwald discusses human information behaviour with reference to three basic concepts: the context, the situation and the social network:

- the *context* is the general setting within which an individual' s interactions take place. Academia, family life, citizenship, clubs etc. are examples of contexts. A context is defined by a set of past, present and future situations.
- a *situation* is a particular setting for an interaction within a context. Teaching a course or attending a committee meeting are examples of situations within academia.
- a *social network* is defined by characteristic patterns and resonances of interaction between individuals within a context. In academia, the social network associated with teaching activity might comprise professors, lecturers, teaching assistants, secretarial and technical support staff and students.

In Sonnenwald' s view, the goals of a collaboration are the sharing of meaning and the resolution of a lack of knowledge condition. For each individual, collaboration within a given situation and context is bounded by their *information horizon*, as defined by the variety of information resources upon which they can draw. In investigating virtual collaboration, there is an important distinction to be made between information resources that can be accessed electronically, and those that are accessed by other means. In effect, each individual has both an information horizon and a *digital information horizon*.

In Sonnenwald' s account of human information behaviour [Son99], each context has its own families of characteristic observables. For instance, in academia, the degree programme, choice of module options and examination marks associated with a student are observables, whilst the weight and height of students are outside the scope of concern. A situation within a context typically includes other pertinent observables that reflect a special focus. For instance, in teaching a course, there is a curriculum, a relevant lecture schedule, and a current point that has been reached in its delivery. Dependencies amongst observables are crucial in shaping the semantics of a situation. For instance, examination marks attained and the current point in the semester may together determine the possible choices of module options, or the entitlement to transfer to another degree programme.

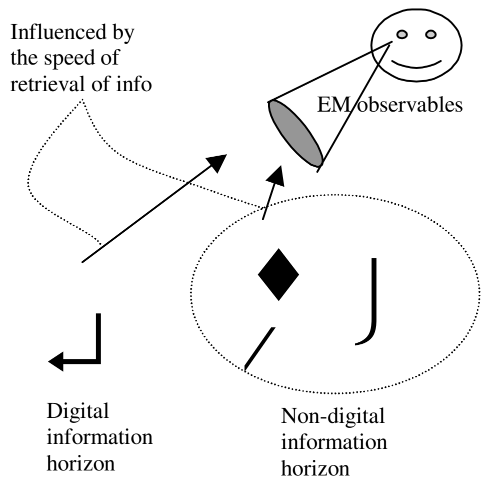

Figure 5.6 Observables and the information horizon

5.2.3 Challenges to virtual collaboration

There are many technical challenges to be met in providing support for virtual collaboration. These include complex and dynamically evolving requirements, as motivated by several key issues [BM00]:

Customisation: Electronic support for collaboration has to take account of the needs of the individual within the social network, situation and context. Identifying and developing algorithms and interfaces to support such behaviour typically requires a high degree of customisation. In particular, electronic support must be well-adapted to the information horizons of the participants.

Integration of the electronic and human activity: Human information behaviour necessarily involves a close interplay between human and automated activity. It is essential that virtual collaboration retains its situated character, so that the information processing activity is appropriately matched to the state of the external world.

Adaptation: The extent to which it is possible and desirable to automate human information processing capabilities is highly dependent upon personal, technological and social factors. The information horizons of participants are typically neither static nor easily preconceived. They can also be influenced in a deliberate way by the actions of participants. Because of these factors, the requirements for a virtual collaboration are subject to continuous evolution.

A major obstacle to successful virtual collaboration is a fundamental mismatch between the roles that humans and electronic devices play in communication and interaction. This is well-illustrated in current practice by products such as document-centric environments for collaborative work. For the human participant, a document is in general full of significance that eludes formal computational representation. Its meanings are rich, ambiguous and contextually determined. In communication about documents, the human interpreter generally exercises discretion, checking the integrity of interpretations with reference to external observation, or feedback from the person or device with which they are communicating. In contrast, an electronic device records and transmits information according to formal pre-conceived conventions, and - if it monitors external state at all - does so in ways that are highly constrained. Static conventions for representing information limit the extent to which the significance of external experiences can be electronically recorded and conveyed. An electronic device is subject to act without discretion, oblivious to its environment. This can lead to catastrophic failure should singular conditions arise.

The traditional approach to resolving these problems of mismatch is to constrain the interaction between humans and electronic devices to patterns for which a very high level of consistency can be guaranteed. As the above analysis has indicated, this approach is not well-suited to the volatile practical demands of effective virtual collaboration. Its limitations are apparent in all three applications a), b) and c) introduced above. Unless these applications operate in stable environments where consistent patterns of interaction can be identified and exploited, the analysis of content and communication in document-centric collaborative work environments is primarily syntactic; the evaluation of user input in intelligent tutoring systems is stereotyped and semantically superficial; automated decision-making in e-business environments is inadequately guided by the high-level interpretation of actions.

Constraining interaction so as to guarantee reliable and consistent responses from devices affects the quality of human contributions to collaborative activity. Experiential and situational elements play a vital part in human interaction. In a virtual interaction, '`no response`' admits quite different interpretations from '`no response`' in a face-to-face encounter. Such issues motivate the integration of different communications technologies, such as telephones, computers and set-top boxes. To explore this integration effectively, it is not enough to view electronic devices and their interaction in abstract computational terms.

The appropriate emphasis is on electronic components and software applications as mediators of state and experience. Successful integration in these terms entails the assimilation of devices and applications into their environment as instruments. Alternative principles suitable for studying automatic agency from this perspective are not only relevant for developing systems to support human information behaviour. They have an essential part to play in the evaluation of environments for virtual collaboration. They can also be used to assess the intrinsic limitations of existing systems and applications. It would be patently absurd to try to integrate computation with batch cards into a fly-by-wire system, but it is more difficult to assess whether, for example, current web and database technology is appropriate for virtual reality.

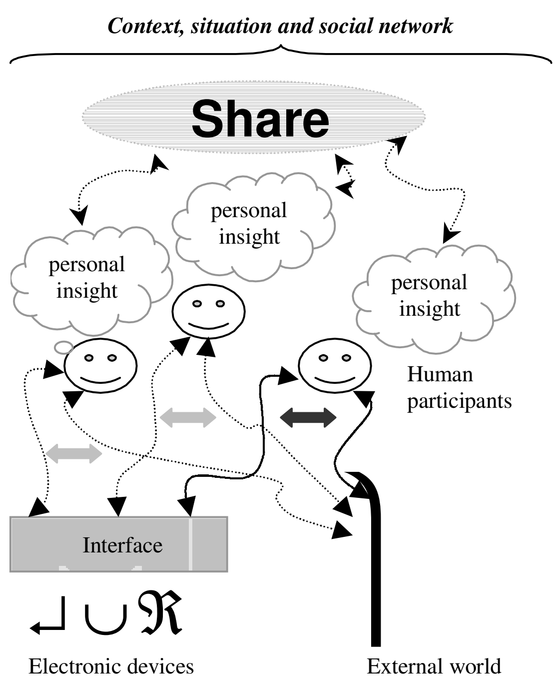

Figure 5.7 Correlating states for successful virtual collaboration

5.2.4 New principles For Virtual Collaboration

Empirical Modelling is centrally concerned with framing and communicating explanations for phenomena. In the context of virtual collaboration, an explanation refers to a convincing way of accounting for perceived state-changes in terms of the interactions of agents. The key questions in this connection are: *What agents are deemed responsible for state-change? What are the cues for state-changing action on the part of agents? What are the direct effects of* *agent action upon the environments of other agents?* Contriving such an explanation requires evidence that is typically gathered from observation and experiment. Subjective and pragmatic judgements are involved in interpreting this evidence. It is not in general possible to give a comprehensive account of a phenomenon in terms of agents and their interactions. Patterns of agency and dependency that can be reliably identified as part of an explanation can be framed as an LSD account. The evolving understanding of a phenomenon that eludes even such partial explanation is captured through developing an ISM. This ISM serves a similar purpose to the physical artefacts that an experimental scientist or engineer might construct in order to express their knowledge of a phenomenon. The ISM can be regarded as representing the phenomenon in the informal sense that experience of interaction with the ISM and with its referent are perceived as having characteristics in common. Creating an ISM is of its essence an open-ended activity in which the modeller can venture to embed ever richer perceptions of observables, dependency and agency. Adopting the terminology introduced by Gooding (1990), an ISM serves as a *construal* of the phenomenon to which it refers.

EM principles can be used for constructing ISMs both as an individual and as a corporate activity. Construction of an ISM by an individual has intimate connections with learning activities [BRSW98], and corporate construction with the growth of shared understanding [SB98]. The roles played by ISMs and LSD accounts in EM represent complementary aspects of experimental activity. The LSD account is a way of framing an explanation; the ISM provides an environment in which to explore and evaluate an explanation. EM activity may involve first framing an explanation in LSD, then generating an ISM as a test environment. Alternatively, it may involve constructing an ISM that can be used to explore possible explanations. In general it is appropriate in EM both to use prior knowledge and to seek experimental insight, and – to this end – to frame LSD fragments and incrementally construct ISMs concurrently.

From an EM viewpoint, sharing explanations and understanding is the key to effective virtual collaboration. In relation to group project work, research in [ABCY94] and [BACY94] have examined the potential advantages of using EM both as a way of reaching consensus in design and resolving conflict in creative partnership and – simultaneously – as a playground for individual experiment. In computer supported education, an ISM can be used both to capture personal insights, and as a vehicle for exploring and communicating understanding (cf. the models of heapsort discussed in Beynon et al (1998)). Current EM research in e-business indicates ways in which EM can be used to investigate how human and automatic agents can co-operate through patterns of workflow and in decision support. The following figure illustrates the relationship between the ISM and the Human Information Behaviour introduced in the previous section. It suggests the construction of Situated Human Information Behaviour Model (SHIBM) based on an ISM that captures: (a) the description of observables representing the context of collaboration; (b) the state of observables representing the situation in collaboration; and (c) the role of agency (introduced as dependencies in the ISM script) in changing the state of observables and establishing a semantic relationship between the digital and non-digital information horizon of individuals. The Human Information Behaviour is implicitly expressed in the interaction with the ISM, and explicitly described in an LSD account.

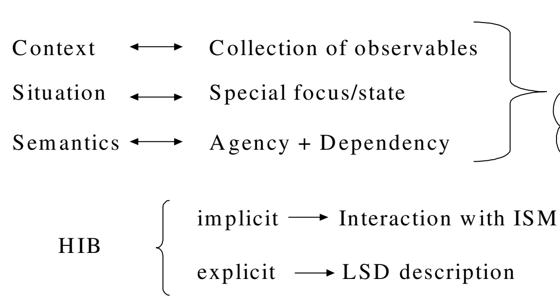

Figure 5.8 A Situated Human Information Behaviour Model (SHIB)

5.2.5 Virtual Collaboration in Online Trading

Different models were presented in [BM00] to illustrate the application of EM principles to various forms of virtual collaboration. For collaborative learning a VEL model (previously introduced in chapter 4) was considered. For collaboration on group projects, examination assessment in the academic context is considered. For business to business collaboration, online trading was considered. This section reviews the latter model of collaboration in the context of a retail trade in NYSE[13].

In modelling an online trading environment, collaboration can be viewed as a workflow[14] of interdependent tasks undertaken by human and electronic agents. A situated human

[13] The story of a retail trade in NYSE is taken from [Har98], and was previously considered in chapter 3 to illustrate the construction of an ISM and the use of LSD notation in EM.

[14] Research in the area of workflow management systems in a business context attributes the difficulty of virtual co-operation between organizations to the lack of standard ways of representing an application's structure and sending and receiving work items [SWH99]. Although the extended markup

information behaviour model of online trading consists of an ISM that supports the exploration of the electronically mediated interaction and communication of human agents involved in the online trading activity. The information horizon of various agents involved in the trading activity is implicitly represented in the interaction with the ISM and explicitly described in an LSD account.

The ISM described in chapter 3 is very simple in nature: it does not take the actual character of the transactions and interactions into account, but merely registers the pattern of the workflow. A more sophisticated ISM would aim at exploring possible scenarios that can arise in the retail trade process (RTP). These scenarios are much more subtle than the workflow alone indicates: in real practices transactions may be disrupted by communication failure, by human error, or by dishonest dealing. The RTP may take place in a setting where other kinds of observation pertain. There will be a stage at which the investor is legally commited to complete, for instance. These broad issues regarding the RTP will have to be reflected in devising a useful ISM.

The potential subtlety of the RTP is mirrored in the possible interpretations that can be given to the LSD account of the broker agent, presented in chapter 3, and the elaborations that these motivate. The LSD account refers to `info_requested` as both a **state** and an **oracle**. This highlights a potential ambiguity concerning a particular information request. As an **oracle**, `info_requested` refers to an observable that is associated with an investor. This can be interpreted as saying that the broker is - or at any rate can be - aware that an investor is requesting information. For the purpose of giving a routine account of the workflow, how such a request is mediated to the broker is irrelevant, and the possibility that the broker may be too preoccupied to note the request is discounted. As a **state** for the broker, `info_requested` refers to an observable whose status is private to the broker. Recording `info_requested` as a **state** potentially admits discrepancies between what the broker believes or recalls and what the investor has declared. The consequences of such discrepancies are implicit in the interpretation of the broker's protocol. The precondition for

---

language (XML) and other similar standards for data exchange, such as Open Financial Exchange (OFX) and Open Trading Protocol (OTP), are significant advances in this area, full co-operation between organizations is still a long way off. Present day workflow systems are not scalable, as their structure tends to be monolithic and they offer little support for building fault-tolerant applications [SWH99]. Software development in the area of workflow management systems is directed towards developing concepts, methodologies, techniques, and tools to support workflow-process management [SAA99]. The main challenge facing the networked economy is to design workflow processes that cross organizational boundaries. This is especially difficult when these boundaries are fluid and subject to continuous change.

action on the part of the broker can be read as: the broker *believes* that a particular stage in the RTP has been reached and that information has been requested.

Similar considerations apply to the **derivate** that determines the stage reached in the RTP. The definition

`stage_in_retail_trade = F(info_requested, … )`

is used to indicate that the current stage in the RTP can be construed as functionally dependent on the status of transactions. In a naïve account of the RTP, this can be seen as reflecting the fact that, once the investor has requested information from the broker, a new stage of the RTP has been entered. Introducing such dependencies in the ISM for the RTP gives the assignment of a new value to the observable `info_requested` the quality of a redefinition - an action that potentially has indivisible effects on the state of other observables. Such a mechanism could also be used to take account of whether an action had some legal consequences, such as might express a commitment or obligation. From this perspective, it might also be appropriate to deem `stage_in_retail_trade` as also dependent upon the precise contents of a transaction: if shares were paid for using counterfeit money for instance.

One motivation for embellishing the LSD account and the ISM for the RTP is that many different communication technologies and information strategies can be used in the RTP. As an observable, an information request placed by telephone has quite different characteristics from a web request. What observables a broker uses to determine the current stage of the RTP may be hard to ascertain. A precise procedural account of how a broker processes a request from an investor might not resemble a redefinition, and could quite easily involve creating and then resolving inconsistent states.

By implication, this is not a simple redefinition, but a sequence of related assignments. Conceptually, it is much harder to guarantee the integrity of the state which it creates. It might on the other hand be necessary to formulate the broker's role in this much detail to capture the true situation more faithfully. For instance, if updating the current RTP status involves some explicit book-keeping on the part of the broker, it is possible for this action to be accidentally omitted.

The above discussion indicates the kind of analysis that accompanies the development of an ISM for the RTP. The precise scope and nature of the ISM is open: it could be oriented towards a high-level account of workflow, or to a specific framework for implementation. The modelling process may be helpful in addressing the integration of human and automatic activities, and could be directed specifically towards related goals. It might also indicate that, in some situations, effective integration is infeasible with current technologies and paradigms, where computation is too far abstracted, and the potential agency of automated components is too limited.

The ISM assists in exploring the ***information horizon*** of human agents participating in the trading process. The information horizon for an investor (cf. Figure 5.9) includes many of the most significant of observables captured in the ISM and described in the corresponding LSD account.

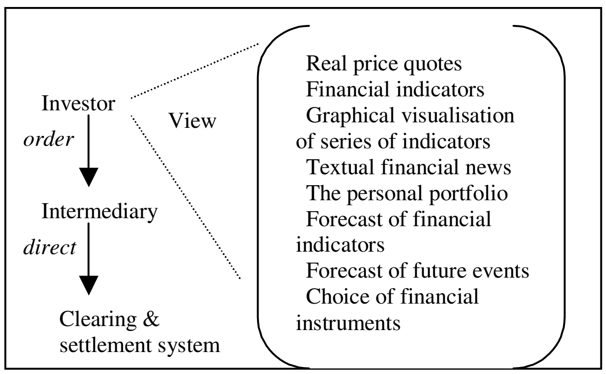

Figure 5.9 A typical information horizon of an investor

Many topical issues motivate the exploration of the information horizon of an investor. Today's investor is looking beyond receiving delayed financial indicators. Trading in a sufficiently liquid and cost efficient market is becoming a major concern for investors [Lan99], and this motivates a better understanding of the trading environment and the layers of intermediation. The support of a large range of instruments, the quality and timeliness of information feed, the functionality of the front-end, and the scalability and performance of the system are important factors in designing digital information resources for an investor. Technology is opening up new avenues for investors to cut out the layers of intermediation and talk to one another directly. This places a question mark over what value can be added to the trading process by the stock exchanges and their constituent brokerages [Lan99]. Current online trading networks provide a huge amount of static information for the investor to interpret and analyse. Online trading web sites have been created by brokerage firms with the aim of extending the digital information horizon of an investor. To this end, these web sites are currently delivering free access to delayed prices, portfolio management services, and graphic visualisation of financial indicators.

5.3 Summary and Future Outlook

This chapter has investigated the potential application of Empirical Modelling technology in the financial enterprise. EM technology can play an important role in the requirements engineering of software system development for the financial enterprise. Integration and virtual collaboration are applications where EM technology can potentially offer a great contribution in directing a special focus to the central role of human agents involved in these applications.

The need and challenges to software integration in the financial enterprise were motivated with the example of integration of e-commerce and enterprise resource planning application. A Situated Integration Model (SIM) is proposed as an integration model that takes into account the social and technical aspects of the software integration activity.

Collaboration is a situated activity that aims at sharing explanation and understanding amongst individuals. Empirical Modelling technology proposes a computer-based support that takes into consideration the human information behaviour and the information horizon of participants in the collaboration activity. Such computer-based support draws on the construction of an ISM and an LSD description that respectively represent the human information behaviour implicitly and explicitly. The complexity of the collaboration activity and the importance of understanding the roles that humans and electronic devices play in communication and interaction is motivated through the discussion of an ISM for a financial retail trade story. Such an ISM would assist in exploring various scenarios and singular situations that might arise in the course of communication and interaction of human agents with programmable components and devices.

The prospect of Empirical Modelling in the financial enterprise is in delivering computer-based support to activities that are human centred and in which singular conditions and changes are likely to arise. With this distinctive quality, Empirical Modelling technology is potentially well positioned to support the development of corporate intelligent networks that integrate the human and technical activity with greater coherence and adaptability to change. Porting EM principles to web technology is another aim that should be pursued in EM research.