Distribution of Integrated Business Applications

This paper, selected for the category “Best papers from 1959 to 2008”, was first published in WIRTSCHAFTSINFORMATIK 35(5)1993:455–464.

DOI 10.1007/s12599-008-0026-z

The Author

Prof. Dr. Henning Kagermann

SAP AG
Co-CEO
Dietmar-Hopp-Allee 16
69190 Walldorf
Germany
henning.kagermann@sap.com

This article is also available in German in print and via <http://www.wirtschaftsinformatik.de>: Kagermann H (2008) Verteilung integrierter Anwendungen. WIRTSCHAFTSINFORMATIK. doi: 10.1007/s11576-008-0134-x.

1 Argument for distributed systems - technology

Value chain integration is a key success factor for most enterprises; an increased service level leads to greater revenues and low stocks improve costs.

In global enterprises, the functions affected are not only organized differently in terms of selling agencies, factories, or procurement departments, but are also geographically diverse. This means it is not only internal organizational boundaries that have to be overcome, but legal and political borders too – a new challenge for the integration of business application software.

The easiest way to solve this integration problem is to adopt a centralized approach. The reality of a global, diversified enterprise can be mapped to an integrated application system that is installed on a central computer, allowing all parts of the enterprise to access up-to-date information directly using remote connections. This physical centralization is also supported by the fact that every transaction is processed in an integrated real-time

network. The entire system is always in a consistent and aligned state. There are no “semi-finished goods,” that is, no semi-processed datasets. It provides the highest standards of integration, consistency, security, and transparency.

A centralized computer system is certainly a suitable solution for an application with well defined performance requirements. However, centralized systems tend to be static and inflexible, something which is certainly not conducive to the globalization process described above.

Moreover, the scarce resources of today’s centralized systems make it very difficult to create user-oriented systems. Consequently, the trend is now to build up dynamically expandable computer networks that allow for changing computing, storage, and presentation capacities.

This solution is called client/server architecture, where services are carried out by specialized hardware and software components in a networked system. The SAP R/3 system has a three-tier client/server architecture, where services for presentation, application logic, and data storage are distributed over different components of a logically integrated computer system. This results in a number of advantages, the three most important of which are discussed below:

Flexibility: Subcomponents that focus on specific business activities can easily be replaced by the cheapest alternative on the market offered by any manufacturer. The system’s capacity can be adjusted dynamically to suit new situations by integrating or removing components from the entire group – for example, for the application logic. It is generally much easier to take full advantage of specific hardware characteristics, integrate new subsystems, and add additional functionality than it is with centralized systems.

Performance: Due to the favorable cost-performance-ratio, end users can benefit

from high capacity reserves. The users are considerably less dependent on the expensive and thus scarce resources of a centralized system. They benefit from the graphical user interface and guaranteed response times for most of their activities.

Availability: Even short-term downtimes can have an enormous impact on an organization; specific applications, such as maintenance, require systems to be available 24 hours a day. Using the client/server concept, multiple computers can work on the same tasks and, if necessary, stand in for a failing computer. At the current time, the database server represents a potential bottleneck, but this will change with the availability of commercially usable distributed database systems.

2 Argument for distributed systems – organizational aspects

The client/server architecture described above allows companies to combine the strengths offered by diverse hardware systems by coordinating heterogeneous system components. To take full advantage of a highly integrated application, organizations also require upgrades that can be carried out on a step-by-step basis as required. This is something SAP is increasingly asked about.

Required organizational or legal changes to a subapplication should not mean that the entire system has to be reorganized. It is not realistic to expect PPC systems that are integrated using inventory accounting and costing to be adjusted because of legal changes to an organization’s accounting application. Aside from cost, this also raises security and feasibility issues.

Feasibility: In today’s world, highly integrated business application software is an enterprise’s lifeblood. Every upgrade – even of unmodified standard software – involves substantial investments in test-

ing, adaptations, interface adjustments, and user training. Moreover, organizational processes often have to be adapted to take advantage of the additional technology provided by the upgrade. These activities are time consuming and consume scarce resources. As integration increases and systems grow, the demand for load distribution and growth at a pace that suits the individual organization intensifies.

Security: All of the activities mentioned above can also be seen from a security point of view. Step-by-step upgrades reduce risk and, because there is less time pressure, the quality of training and testing improves.

3 Client/server applications

Consequently, exploiting the advantages of the client/server model requires two changes: moving from a centralized computer to a network of cooperative, powerful hardware units and splitting the application into multiple, largely decoupled units without sacrificing important integration benefits.

Central services play a particularly important role in this context. For example, the basis component, which in accordance with the layer model hides details of the operating system, database, communication, and presentation software from the actual application and the central application units. The latter guarantee a consistent use of specific methods such as classification, documentation of changes, calendar functions, and the management of units of quantity, currency, and time.

There are essentially three approaches to meet the requirement for step-by-step upgrades. They also lay the foundations for distributability at a later date.

3.1 Remotely coupled complete systems

At first glance, the easiest option may seem to be the substitution of the central application with multiple remotely coupled complete systems that have specific application functions. This method has been frequently applied by SAP software users using standardized interface techniques. There are no systems engineering prerequisites and coupling two releases of the same system is much easier than it is to couple third-party systems. Even where users have high performance requirements

A 3119 D X0 18092

TWS = Treasury workstation	DOCS = Decentralized order control system
FAM = Financial assets management	RF = Financial accounting
EIS = Executive information system	RA = Asset accounting
CAS = Computer aided selling	RK = Cost accounting
WMS = Warehouse management system	RK-P = Projects

Fig. 1 Decentralization through satellite systems

(replication of master data or remote availability tests), this option appears to be a practicable solution because it allows organizations to avoid the difficult process of splitting the system into disjointed and independently upgradeable application components.

However, there are a number of disadvantages to this approach, which have less to do with the increased number of resources required than the restrictions on the users to centrally run, if required, such subapplications that are upgraded at varying time intervals. Moreover, the method does not help reduce complexity and therefore does not help an organization either control or maintain highly integrated systems.

3.2 Satellite systems

To keep the core of a highly integrated system lean, an organization can gradually outsource special functions of a defined scope and which do not have to be regularly synchronized with the core application.

Delegating special peripheral tasks and decentralizing computer units in this way facilitates load distribution and downtime security. The increase in local autonomy saves time-consuming and costly access

to the central computer. Satellite systems can also temporarily work independently of the mainframe, while being integrated with the respective centralized application by means of program-to-program communication to allow them to react quickly to internal and external events.

Fig. 1 uses the SAP R/2 system to illustrate the multitude of tasks that may be decentralized in this way. These include:

- Decentralized warehouse management and decentralized shipment,
- rule-based product configuration,
- periodic shutdown processing,
- computation-intensive development of forecasts and statistics,
- executive information systems and decision support systems.

In this way, a complex centralized control instance can be split up into smaller, clearer elements. Information is processed more quickly and directly on-site, with a high degree of flexibility and fast reactions to local factors, as the following examples show.

Flexible production: At factory level, the entire supply chain is supported, from production, planning, and order processing to material management and capacity planning. Distribution rules control the transfer of completely compiled order and master data to control stations (DOCS) for the

individual production areas. These control stations autonomously process their worklists and communicate with each other using defined messages. The CPI-C protocol (Common Programming Interface – Communication) is used to connect to a number of different subsystems at cell level by means of communication channels with specific functions, for example, for PDA/MDA, transport/warehouse, and entering quality data.

Cash management: Optimal cash flow management in terms of risk and return requires the real-time integration with the process steps ordering, goods receipt, invoice receipt, or ordering, goods issue, invoice issue. However, a lot of cash management tasks can be pre- and postprocessed offline by autonomous computers. Such tasks include:

- Communication with banks and financial markets,
- financial investments and loans in the case of a short-term liquidity surplus and shortage,
- monitoring of foreign currency positions and currency transactions.

Satellite systems offer a smooth transition from the SAP R/2 to SAP R/3 application. Each satellite system has its own separate database that contains the necessary master and transaction data and messages from the central SAP system. The distributed data storage is hidden from the end user. The SAP software guarantees the consistency of the decentrally stored data.

The system selects the appropriate protocol for the program-to-program communication between the satellite application and the central SAP R/2 system. This protocol may be CPI-C for synchronous links, Q-API (queue application programming interface) for asynchronous communication, or Remote Function Call and OSFIRPC (Open Software Foundation's remote procedure call) to call functions on remote computers.

3.3 Application decoupling

In addition to distributing to different computer units, replacing defined parts of an integrated application system with new releases also requires the software to be segmented into autonomous, cooperating modules. First, it is necessary to map single software objects (for example, data elements and function modules) to subapplications. This task is very important

in the case of highly integrated systems, where the consistency of the business solution depends on – in addition to integrated data – the consistent use of standardized methods, that is, the use of identical function modules in diverse applications. In this case, it is important to find a compromise between multiple use and decoupling. If the number of identical objects becomes too large, this can result in inflexibility (global objects are either not to be changed at all or only infrequently) or increased development costs (global objects are only to be changed if downward compatibility can be ensured). For the same reason, it is necessary to define stable APIs that are as lean as possible that also belong to the global objects.

These measures are all essential prerequisites to run application components on multiple computers. If the APIs are called using remote procedure calls (RPCs), nothing can prevent distribution. Nevertheless, the influence of the network on response times must be absolutely minimal and any possible transfer errors must be negligible. At the same time, the availability of all computer nodes has to be assured.

Client/server systems that distribute the application load to different computers require powerful networks. In the case of large applications, this currently implies a restriction to the local area network (LAN) area and the use of the same processing methods as those in the centralized system. The dialog is purely synchronous, which means that the client waits for the service to answer its request before continuing processing. In the booking part, all secondary activities can be performed asynchronously that provide applications with non-critical status requests. These include, for example, updates of statistical files that can only be evaluated periodically or updates of totals record files that are carried out on the basis of high performance evaluations. If this is not the case, additional rules have to be defined and implemented that define how individual transactions must react when faced with specific faults, which replacement transactions are to be triggered, or which follow-up activities are to be carried out. It is precisely this handling of exceptions based on real-life situations working together with the user that makes up a significant part of successful standard business software.

In this context, the low cost and high performance distribution of integrated

systems without any application knowledge is neither realistic nor desirable. When the application is designed, the extent to which consistency and timeliness of the data are required is evaluated to guarantee integration benefits. Not only does this promote the system's flexibility, but it can also prove more advantageous to have a certain level of redundancy in specific applications, where data is slightly less up to date and inconsistent in the short term. The requirements cannot be fulfilled by distributed databases, but instead require distributed transaction processing in which processing orders and results are exchanged between cooperating subapplications.

4 The argument for distributed systems – economic aspects

Globalization means the diversification of markets and products, the frequent acquisition and sale of parts of companies, consolidation, and strategic alliances. An integrated application has to cope with an emerging organizational heterogeneity and hide this from the user as much as possible.

Changes arise too frequently to use consistent key terms, key systems, and business processes to create an organizational framework for a logically centralized system with synchronous and direct communication methods. The distances between an international enterprise's various locations around the world are so great that the effect of slow and error-prone transmission media (WAN) must be taken into consideration.

Consequently, locality and step-by-step growth become more significant and not simply from a technical perspective. Local processing not only reduces communication overhead, but also enables companies to adapt to local organizational conditions. A distributed application matches the organizational structure and management principles of a global enterprise. It should be possible to dynamically adapt distribution models (client/server, master/satellite), communication methods (synchronous, asynchronous), and the form of the coupling (loose, strong) to general conditions; the business methods (selection of the evaluation strategy, creation of contingents) must be appropriate.

5 Organizational forms and distribution models

It is possible to distinguish between three ideal cases:

Strategic alliances (for example, to reduce vertical integration) require information to be exchanged smoothly when processing intercompany supply and output obligations. There is less need for integration between distributed applications. Integration is only required for the logistic quantity flow and external accounting.

Federated groups (financial holdings) are unions of loosely cooperating enterprises that share the same management. The subsidiaries are considered to be financial investments; the need for integration is restricted to the finance function only.

Centralistic groups (management holdings) represent economic units in spite of national and legal differences. There is an extremely high level of internal supply- and output-related transactions and an unlimited demand for integration, even across national borders.

5.1 Intercompany communication with EDI

In a strategic alliance, business partners must be integrated to ensure that the entire value chain is optimized. This also requires the partners' different software systems to be integrated to a certain extent. The advantages of the technical solution electronic data interchange (EDI) have already been proven and enable, for example:

- The gaps between heterogeneous data structures to be bridged using transformers,
- individual transmission formats,
- diverse communications strategies.

However, even in this simple scenario of coupled systems, the application software has to be prepared for EDI; the greater the degree of integration within the applications, the greater the extent the software has to be prepared for EDI.

For example, an outgoing invoice is sent out to the customer using EDI technology. When the customer receives it, it is automatically transformed into an incoming invoice. Since the customer's cost accounting system is integrated with its accounts payable accounting system, the account assignment data that is stored within the order must be added to the incoming

Fig. 2 Multilevel consolidation: company - subgroup – group

invoice. To meet the partner's integration requirements, the supplier's order processing must be able to transfer the customer's order number through all the steps in the process until it reaches the outgoing invoice so that from there it can be included in the EDI message.

Moreover, EDI can be used for internal communication between a company's local applications, for example, where internal services have to be billed using summary billing documents for legal reasons. EDI can be considered an option if coupling only applies to posting and data is communicated asynchronously by means of wide area networks (WANs).

5.2 Master/satellite model

The need for integration between local applications in federated groups is comparatively small. Usually, this demand can be satisfied by modern consolidation software (for example, the SAP R/3 component FI-LC). The external reporting example (see Fig. 2) shows the major transformation steps necessary to harmonize information from heterogeneous supplier systems and aggregate it in a multilevel cross-computer manner. Incorrect information often prevents a group from preparing its financial statements on time. Consequently, this technology also requires the local subapplications to be prepared for consolidation at a later stage.

Due to the fact that a comparatively smaller amount of information is exchanged at a few defined points in time, the choice of the communication medium is not as important. EDI can be considered optimal, while data medium exchange is also usually sufficient.

In these conditions, this form of integration of distributed systems can also be

transferred to other areas. However, it is not suitable for planning processes where decisions must be made on a case-by-case basis, but can be applied when centrally coordinating largely autonomous units. To enable decentralized units to be run commercially autonomously between the coordination phases, it is essential to disaggregate central specifications in terms of budgets or contingents. The cooperation is based on the master/satellite concept:

- Specification of the reporting structure, which is to say, reporting formats or central key terms are transferred to the divisional computers;
- Reporting of the area's resource requirements to the headquarters – in an aggregated form and in accordance with the steps illustrated in fig. 2;
- Distribution of approved budgets to the divisional computers;
- Detailed control and scheduling of corporate divisions using real-time integration on the divisional computers;
- Periodical reporting of aggregated information to the central computer.

5.3 Distribution of highly integrated applications

Many international groups are currently undergoing a phase of increased decentralization to improve flexibility and customer focus. However, at the same time, there is an increased need for business processes to be integrated to stay competitive. The scenarios addressed above do not provide a sufficient solution for such cases.

SAP develops potential solution models using pilot projects based on real business cases. The following steps are used:

- identification of typical distribution patterns in different organization types;

Fig. 3 Scenario of a distributed organization

- Evaluation of the information exchange by frequency, amount, and reliability;
- Determination of distribution principles and communication forms (multiple options if applicable);
- Adjustment of business methods and processes to correspond to the distributed information structure.

Fig. 3 shows an ideal configuration pattern with which the decentralization is aligned with the organization of the activity output process. The choice of location and organizational structures for sales, production, and other central services (R&D, maintenance, and so on) is specific to the individual functional areas. The coordination and responsibility for earnings lie with the individual divisions, although the management of the division does not have to be based at the same location as one of the areas. The aim is to

achieve synergies using central functions in finance and procurement and avoid the suboptimization of the divisions by means of group controlling.

The scenario depicted in **Fig. 4** and **Fig. 5** assumes that the system is distributed in accordance with the organizational structure and that each location runs its own local application on a computer or computer network. Cooperation is based on different principles depending on the application's special requirements.

5.3.1 Logistics planning example

If you look at a division from a logistics planning point of view, a two-tier master/satellite construction is clear (**Fig. 4**):

- Central sales and production planning at division level coordinates the sales

and production activities of the decentralized areas.

- Worklists of production orders are distributed to control centers by the production area's PPC system. Here, they are processed asynchronously and remain inaccessible to the PPC system. Once the technical completion has been reported, they are returned and further processed, if necessary (see section 5.2). A similar structure exists for coupling the distribution sector with the distributed warehouse management systems in the distribution warehouses.

Horizontal coupling between functional areas is much more time critical and is carried out in accordance with the client/server principle. Since the aim is to achieve process integration (see section 2), it is essential to start processing specific orders of individual clients immediately.

A frequently occurring order type is the validation of shared master data, for example, product, customer, account, and the substitution of deferrable key terms. The authorization to maintain such data can be assigned exclusively to one subapplication. Moreover, master data is changed relatively rarely and, if it is to be changed, this does not take effect until the next day or even the next month. There is therefore enough time to distribute read-only views using a copy management function to all possible nodes and thus reduce network load.

Another critical order type is availability control. If there is a shortage of a particular item, an online check must be carried out immediately to establish whether the required stock is available at other locations or when it can be produced. This is essential – particularly when the items concerned are not frequently in demand – and cannot usually be avoided by local product allocations. In such cases, availability is checked by means of remote procedure calls. The amount of data transferred is relatively small, and slightly longer response times can be tolerated.

5.3.2 Example of profitability analysis

Internal accounting is based on the structures of the activity output process and thus can largely be carried out autonomously on the nodes of the configuration depicted in **Fig. 3**. Cooperation is based on the following principles (**Fig. 5**):

- Data redundancy: Local account assignment checks are ensured using

duplicates of the relevant master data and local valuations by means of a periodical exchange of the valuation files

- Remote procedure call: synchronous availability checks for central budgets
- Distributed posting: asynchronous updates in the local system of the data that is recorded in a central accounting system and is relevant for cost accounting
- Master/satellite technology: Complete objects, such as cost centers, cost objects, and market segments, are transferred periodically to the master system where they are processed.

6 Improved distributability through organizational measures

Restructuring business processes and business methodology is often essential to release the full potential of integrated application software. Many enterprises consider this a positive point. This willingness to make small changes to the company to gain flexibility, performance, and autonomy is also required for distributed systems. The three different cases below offer examples of such adaptations:

Aggregation of information: A substitute activity is triggered by temporal or logical conditions that aggregates information produced by various initial activities. Consequently, certain activities performed by the sending system are only offered to the receiving system indirectly in an aggregated form. A typical example would be summarized periodical postings to financial accounting accounts, triggered by cross-company activity allocation, which are, however, reported in real time in the cost accounting systems.

Quota arrangement: A central coordinator defines general conditions using quotas or budgets within which the areas can act autonomously. The more time-consuming and performance-intensive global availability check is only initiated if the local check indicates a lack of availability.

Processing when global information is missing: Integrated processing in a centralized system ensures all the required information can be accessed. In a distributed system where components only communicate by exchanging messages, it takes a certain period of time before information that is not available locally can be accessed. Furthermore, complex lock-

Fig. 4 Logistics planning in a distributed application

Fig. 5 Profitability analysis in a distributed application

Abstract

Henning Kagermann

Distribution of Integrated Business Applications

New corporate developments, such as globalization, diversification and process orientation, are posing a challenge to the degree of integration of business application software. International standard software designed to meet the requirements of all types of business is acquiring greater strategic significance. The increased need for flexibility, localization, and scalable growth makes it necessary for highly integrated businesses applications to be distributable. Possible scenarios reflect managerial and organizational requirements. Additional mechanisms must be incorporated and business processes adapted to compensate for the increased autonomy of sub-applications.

Reprint of an article from WIRTSCHAFTS-INFORMATIK 35(5)1993:455–464.

Keywords: Standard software, Distribution, Integration, Corporate management, Business applications, Client/server, Master/satellite

ing mechanisms have to be implemented, which tend to hinder the application.

Therefore, it is more practical to solve the problem iteratively. The triggering sub-application, the client, processes an activity using the locally available information, based on the assumption that it has sensible, representative information about the partner applications that are also involved but operated on different servers. If applicable, the results of the servers trigger automatic corrections of the initial response to the client. This approach is in line with tried-and-tested business practice for the valuation of connected but temporally separated events based on cost-accounting assumptions. The only disadvantages are for the software manufacturer that has to pay for the gained practicability with significant additional effort and costs in application development.

Let us take the example of stock materials procurement in a configuration where materials management runs in a different system to financial accounting.

Goods receipt is processed in the material management system. All information is available locally to correctly update the quantity and value and the material stock (based on the agreed purchase order price). A deferred update of the general ledger does not result in corrections.

A few days later, the incoming invoice is received and recorded in the financial accounting system. The amount is higher than that assumed at the time of goods receipt. Local information is now missing from the purchasing and material management system to enable the postings to be executed correctly. Is it possible to accept the invoice objectively and release it for payment? Is the quantity of stock large enough to subsequently debit the stock account with the difference amount? A possible solution would be additional clearing accounts, where postings can be held temporarily until exact global information about the business process is available. Since timeliness is more important than ultra precision in controlling systems, one suitable solution would be to assume the postings are executed under normal conditions (for example, the items are in stock, invoices that have not been released are to be blocked). Automatic repostings are only to be triggered if the receiving system determines any deviations from these normal conditions.

One apparent disadvantage of this procedure seems to be the temporary inconsistency between subapplications. This is

insignificant, however, if at any time an online trigger can force all open information orders to close, establishing a consistent state.

7 Conclusion

The scenarios for the distribution of integrated applications are shaped by business and organizational requirements. Distribution models and forms of communication have to meet the application's needs and should be configurable to a limited extent.

The application must be structured in such a way that it increases the autonomy of the subcomponents. This has to be compensated by additional, practical mechanisms and adjusted business methodology. When SAP replaced methods based on periodical batch processing by dialogue-based software, it was not a problem to, for example, replace an average price calculated on monthly basis by a floating average price suitable for real-time processing. Similar adjustments can also be expected with the transition from centralized to distributed application processing.

Without having to sacrifice any integration advantages, users gain a high degree of flexibility as regards:

- Their ability to react to changed conditions;
- Their ability to adapt to new markets, new organizations, and new technologies;
- The scalability of their systems and procedures.

Despite increased modularization and standardization, the software manufacturer has to overcome an increase in the internal complexity of its products.

References

- Coulouris GF, Dollimore J (1988) Distributed systems – concepts and design. Addison Wesley
- Jablonski S (1989) Datenverwaltung in verteilten Systemen. Springer, Heidelberg