Inter-database synchronization: a low-cost approach to information system integration

Fajar Suryawan
Department of Electrical Engineering
Universitas Muhammadiyah Surakarta
Surakarta, Jawa Tengah 57102, Indonesia
Email: Fajar.Suryawan@ums.ac.id

***Abstract*—In this preliminary report paper we discuss a low-cost approach to information system integration ("ISI") in an enterprise setup. The approach relies upon the notion of *unique identifier* (UID) and continual synchronization. The approach is low-cost since it does not require high-skill from the IT staff (as compared to, for example, enterprise-service-bus approach), relatively easy to develop, and can be built entirely on well-known open-source software packages.**

The approach can, and should, be regarded as an intermediary technology before a company is able to deploy and operate a full-fledged integration scheme for its information systems. The research is based on an ongoing design and study of ISI in a university setup.

***Keywords:* information systems integration, data quality, database design, data synchronization**

I. Introduction

The computer era has brought an unprecedented change in the way people perform business. What once a one-day paper-based query is now just a matter of seconds. In short, business is performed over electronic switches. The computer revolution has been well embraced by the more vigilant companies: by adapting to the new technology, re-engineering their business process, or becoming a new kind-of-business altogether. That is, IT becomes an enabler. Many articles and monographs have been written about this track since the last decade of previous century [1]–[7].

A decade later, value of IT for business is still, and even more, an active research field [8]–[11]. Many undergraduate texts have incorporated the IT value for business as one of their main topics. See, for example, [12].

However, on the other side of the statistics, some companies are still struggling with a number of IT-related basic issues. In these companies, IT is treated just a bit more than a way to automate business processes. In effect, the benefits derived from the new technology are sub optimal. The situation is often corroborated by several factors, including 1) poorly designed legacy database, 2) disparate information systems spread across organizations within a company with virtually no connectivity, and 3) virtually nonexistent coordination between administrators of those information systems. This is even truer for organizations that have not successfully adopted an IT governance framework, have relatively young IT history, or simply do not have the necessary resources to maintain the complexity of the IT environment. The circumstances has led to practical problems such as

- data redundancy,
- non-existence of single source of truth (SSOT) to refer to,
- continual effort of data synchronization (if ever), and
- lack of executive confidence in regards to the data and information quality.

These problems are very typical in *all* universities in Indonesia (even the biggest institutions and especially the smaller ones), and there is even a national-level effort to address this issue.

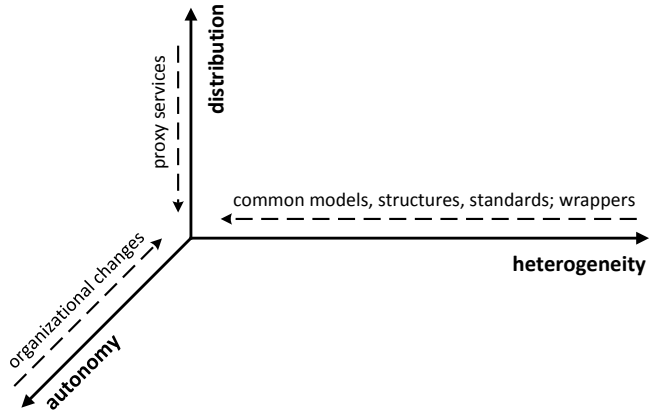

Fig. 1. Three problem dimensions for systems integration: autonomy, heterogeneity, and distribution. The dashed arrows indicate some general approaches to manage these issues. From [7].

Several works have been written about information system integration. Some are from managerial point of view, and others are from the more technical one.

Hasselbring [7] explains an overview of information system integration, including IT disciplines involved in ISI and dimensions in ISI, see Fig. 1. In that article's framework, our work here is to build proxy services (the 'distribution' dimension) and to impose wrapper and common model (the 'heterogeneity' dimension). In the 'autonomy' dimension, our approach asks for several minor organizational changes.

The work by Wagter et.al [13] opens several insights, one of which is the pervasive problem of tension between Agility and Coherence. In aligning business objective and IT solution, two "forces" are influencing the process in almost diametrically

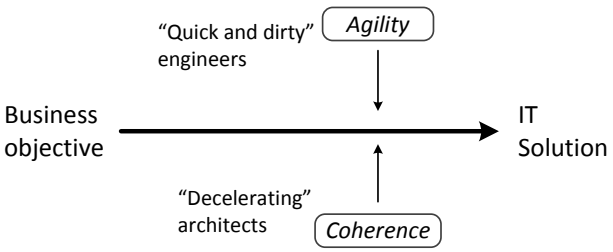

Fig. 2. Tension between Agility and Coherence. From [13].

opposite way: Agility (practiced by "quick and dirty" engineers) and Coherence (practiced by "decelerating" architects), see Fig. 2. This problem emerges as organizations are redesigning their business processes in an increasing rate. (In 1970s to 1980s it was once every 7 years, in 2000s it can be once every 3 months, see [13] page 15–16.) This phenomenon also happens in Indonesia, and the IT departments are increasingly difficult to catch up with the changes. In the spirit of the book, our work here attempts to balance the two forces. We allow developers to work in agile manner, but do so within specified architecture (in this article, mainly database-concerned).

Zarnekow et.al. [14] describes an IT management strategy using methods and best practices learned form industry world. The monograph by Ulrih and Newcomb [15] provides guidelines and case studies of IT modernization. Governor et.al. [16] gives a constructive overview on how an information architect should utilize and profit from Web 2.0. This is important since web-based interaction is increasingly ubiquitous.

The above works provide strategic guidelines on how to run enterprise information systems, especially in the scope of information systems integration. However, it is always up to each organization (and each architect) on how to develop and administer the changes in the tactical and technical levels. A cursory survey follows.

Fu Xiaolong [17] reports (based on a case study at Tsinghua University, China) a strategy to develop a platform for university-level information system integration. The university utilises, among others, a service bus and unified authentication/privilege systems. Citrix software package is used in the integration scheme. Fong [18] details a prescription for information systems reengineering and integration. Williamson et.al. [19] elaborates an Oracle-based information integration, migration, and consolidation. Ding and Lin [20] describes information architecture mainly from user experience's point of view. Bonnet et.al [21] illustrates way of overhauling information systems with service-oriented architecture (SOA).

While some of the works are detailed and solution oriented, it is implicitly assumed that the brainware having the required skill set is available. In this paper we attempt to address the problems above by using the typical resources available in a small Indonesian university.

The rest of the paper is organized as follows. We first outline an analysis where we describe the setup, assumptions, and requirements. In Section III we will explain the design, after which a discussion section follows. A conclusion section closes this article.

II. Analysis

A. Background and Setup

Indonesian private universities are typically under-resourced in technology-related talents. When a university develops an electronic information system, the database design are most of the time done in a hurry and without full thought for future needs. Sometimes the resulting databases suffer from non-normal tables and smartkey problems. Lack of talents also means that no coordinated database effort occurs in the university-level, and there is no list of enterprise main data elements[1].

In effect, each office and faculty (e.g. HRD, Research Office, Faculty of Engineering) in an institution develops its own, almost totally separate, administration system. What to come next is already expected: an entity is present in several places, and there is no single source of truth regarding an entity.

B. Resource Assumptions

Given that a number of information systems *do* exist in the university of concern, it is reasonable to assume that talents are available with the following skill set:

- HTML, CSS, JavaScript, and PHP Programming
- MySQL or MS-SQL database design and maintenance
- Linux or Windows Server administration
- OpenLDAP or ActiveDirectory administration

Skill items number 1, 2, and 3 are very common throughout Indonesia and are generally what is taught at college-level web-design related courses.

Skill item number 4 is admittedly rather rare. However, at its basic functional level, this is a relatively straightforward technology, and simple training or induction should suffice.

We should also mention the other side of the story, that the university of concern suffers from the following.

- Generally an enterprise architect is not available
- COTS solution, such as SAP™, is generally not an option. This is due to affordability issue and lack of talents for its operational.
- Existing programmer are generally have high workload already.
- The network infrastructure is not reliable.

C. Requirements

In spite of the glooming existing condition, one should strive for order. The following is a list of requirements for the better final state.

- There is one list of main data elements, including who has the privileges to create-read-update-delete (CRUD).
- There is a single source of truth for every data element.

[1] Younger universities however, learning from the mistakes made by the older competitors, have the advantage of being able to design the system from scratch.

HRD.staff

id_pk	Em_num	UniID	Name	Addr	Pay
7200	100.302	ka355	Ken Arock	Jogja	5500
7201	500.334	js123	John Smith	~~Jogja~~ Klaten	4560
7202	335	pa776	Pat Adams	Solo	4280
...	...	...	...	...	...

Research.people

id_pk	UniID	Name	Addr	NPub	IEEE
180	hr198	Hari Rimau	Sragen	23	M
181	js123	John Smith	Jogja	44	F
182	ja232	Joe Abrams	Jakarta	21	SM
...	...	...	...	...	...

Fig. 3. An example of update mechanism with UniID as the synchronization key. John Smith changes his address from Jogja to Klaten through an HRD interface. Research Office will also need this update, so when the cron-job executes, it scans for updated records. Then using UniID as the pivot, the target table at Research Office is updated. Not every attribute in `HRD.staff` is copied to `Research.people` (e.g., the `Pay` attribute), and the target `Research.people` table can have its own attributes not present in the source table (e.g., the `NPub` and `IEEE` attributes).

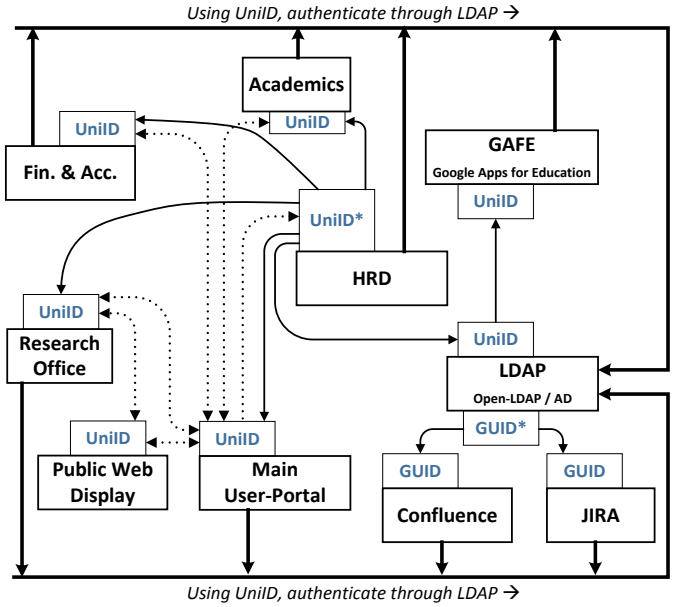

Fig. 4. An example of inter-database communication architecture in a university. Solid thin lines indicate cron-job procedures (not necessarily having similar timings), dotted lines indicate *plain* web-service based communications. Solid thick lines show authentication track. It is important to note that in developing synchronization procedure, one can use web service technology as well.

- Typical information retrieval can be done in a matter of seconds or minutes, and more special ones can just wait for 'next sync'.
- Given a clear query, the retrieved information is exact and unambiguous.
- Programmers can still develop softwares using their existing practice, and only minimal modification takes place.

The above requirements necessitate several practical needs in the database layer, such as database normalization and CRUD matrix detailing. Some systems may even need database reengineering. While those are very big topic by themselves, in this paper we assume they are done already.

III. Proposed Design

A. Principal Mechanism

In the proposed approach, each information system is, as before, developed independently. These information systems can have their own databases, tables, and the pertaining primary keys. The important difference is that now, specific entities in the database have to be synchronized with their counterpart from an 'authorized' database. An example follows.

Entity `staff`, which resides in `HRD` database schema, has attribute `name`, `address`, and `title`. If someone changes his address, this is the table in which the relevant record is updated (through some interface, of course). One's title can be changed but not by him, instead by an HR officer. Here we say that `HRD.staff` is the *single source of truth* for staff data.

Other information systems (e.g. Research Office, Academics, Public Web Display) may also need to have the `staff` entity. Previously, this was done by 'cold copying', and then each of those same entity within different information systems took life on its own. And even worse, there was no designated source of truth.

Now, with the new approach, those other information systems are required to *synchronize* their `staff` entity with the `HRD.staff` table. This is done using the following strategy.

- In the `HRD.staff` table (the staff data SSOT), we insert a column named ***`UniID`*** (will be explained later). This unique attribute is *not* the primary key of the table. Rather, this is a *synchronization key*.
- Other information systems (e.g., Research Office, Academics, Public Web Display) may have their own staff table with any of the attributes (and with their own primary key designation system). It is required that these tables must have the `UniID` column.
- At pre-specified times, these other information systems update their staff table to reflect the changes in the `HRD.staff` table (in some pre-specified columns). For example: Research Office updates its staff table every one hour, Public Web Display updates every 4am daily.
- The update takes place with `UniID` as the reference point. Any change in the record for a particular UniID (that is, a row having that UniID) in `HRD.staff` will be reflected in the row with the same UniID in all staff tables in each database. Fig. 3 illustrates this.

This synchronization ('sync') scenario is performed for several information systems, as shown in Fig. 4 (solid thin lines/curves). Plain web services are also used to access databases of other information systems, as depicted in Fig 4 using dotted lines. We refer this as 'plain' web services since the sync method above, if so decided, can also run via web service technology [22],

[23], [24].

The sync method is chosen over plain web service one if the internal mechanism of the system of concern requires heavy access to a staff table. Hence in the case of connection failure (so the current-time update does not happen), it can still run smoothly. On the other hand, the plain web service approach is chosen if an information system only needs occasional access (read or write) to the SSOT table. For example, the Main User-Portal in Fig. 4 uses plain web service to allow a staff to change his/her address.

B. UniID

UniID, an abbreviation of the phrase 'University Identity', is a person's attribute. It is used as synchronization key and username with which one logs in to the institution's information systems. It has the format that in general read "first characters of the words in one's name, followed by three randomized digits. For example, 'John Smith' might have UniID 'js123'. In Fig. 4 it is shown that the authentication process is done through an LDAP system (e.g., OpenLDAP or Microsoft Active Directory).

The idea of UniID resembles that of GUID (Globally Unique Identifier) or UUID (Universally Unique Identifier). It has big name spaces, unique and meaningless (so one, ideally, would not bother to change his/hers), yet easy to remember. It also has the advantage of complying to a number of older systems which accept only equal-or-less-than 8 digit usernames. As for email addresses, a 'default alias and send-as' system is employed.

Several institutions are using this scheme. We adopt the name 'UniID' from University of Newcastle, Australia which, as far as our knowledge goes, uses similar usernaming scheme for its staff.

There are other systems, usually COTS ones, that more suitably connected through LDAP system. For example, Confluence software (of Atlassian Inc.) has native connection through OpenLDAP or ActiveDirectory. When the authentication is done using an OpenLDAP server, it uses OpenLDAP's GUID as the synchronization key. Any change in a person's record in the OpenLDAP system (even the username–UniID in this case) will be reflected in the Confluence system. This is shown in Fig. 4 in the lower right corner.

C. Design Decision for New Software

A new information system may be in order. To maintain some degree of data integrity, we need to ensure that the new application conform to the new method. The following is a list of questions for the analyst to answer in their design.

- Does it need staff record? For read only or also write?
- If yes, which columns from staff table does it need to access?
- How crucial is it? Will it be using plain web service, or synchronization?
- If synchronization is to be used, what is the schedule? Every one hour? Every 2am?

Once these questions are addressed, software designers and developers can begin their work.

D. Technical Issues

If one is to implement this architecture, the following issues have to be addressed

- The place for the cron-jon procedure. Is it in each requiring software, or is it in one place that 'pools' all the cron-jobs?
- The method to speed up synchronization. For example, using time-stamp to detect latest record updates.
- Procedures to anticipate and mitigate disruptive mishap occurrence (e.g., power break down in the middle of sync-ing).
- Procedures of maintenance sync-ing (e.g., once a week).

IV. Discussions and Insights

The approach taken here should not be regarded as final nor of best practices one. It attempts to tackle the problem of ISI using typical available resources in an Indonesian university. Below is a guide on how to start to proceed from this state.

Basically the main issues faced with the university of concern are that of IT Governance and [human] resource management. To this end, it is our recommendation that any institution facing a similar problem is to,

- on the strategic level: engage a business-and-IT alignment program within a framework, such as COBIT [25].
- on the managerial level: attract, hire, and keep the best talents, and practice better talent management.
- on the technical level: start exercising best practices on every level as much as one can. In this ISI topic, one can research on, for a start:
  - database engineering best practices: CRUD matrix, normalization, surrogate keys, etc.
  - data management best practices: data warehousing, privilege management, etc.

The proposed solution is aligned with a number of COBIT 4.1 points. The more technically relevant points are discussed below.

- **PO2 Define the Information Architecture**. The proposed solution requires an organization to explicitly list main data elements and its access management.
- **DS5.3 Identity Management**. The proposed solution inherently has a key used as the identity token with which a staff is logging on to the organization's information systems.
- **DS5.4 User Account Management**. The proposed solution requires an administrator to manage user accounts with an LDAP system, which is synchronized with an SSOT for staff data.

V. Conclusion

In this paper we proposed a low-cost approach to information system integration in a university. The method utilizes GUID-like synchronization key called UniID. Each information system database requiring staff record must synchronize with university-wide single source of truth regarding staff data. Plain

web service based communication is also used in conjunction with the sync method. The method is low-cost since it requires only typical skill set available in a small Indonesian university.

ACKNOWLEDGMENT

This research is based on still-ongoing ISI project at Universitas Muhammadiyah Surakarta (UMS), Indonesia. The author would like to thank the UMS management for their full support for the research.

The author would like to thank the reviewers for their constructive suggestions. The author also would like to thank the HELM project from USAID, run by Chemonics, which facilitates a forum for a number of Indonesian universities in the effort to tackle the information system integration problem.

REFERENCES

[1] M. S. S. Morton, *The corporation of the 1990s: Information technology and organizational transformation*. Oxford University Press, 1991.

[2] T. H. Davenport, *Process Innovation: Reengineering Work Through Information Technology*. Harvard Business Review Press, 1993.

[3] J. C. Henderson and N. Venkatraman, "Strategic alignment: Leveraging information technology for transforming organizations," *IBM systems journal*, vol. 32, no. 1, pp. 4–16, 1993.

[4] T. Mukhopadhyay, S. Kekre, and S. Kalathur, "Business value of information technology: a study of electronic data interchange," *MIS Quarterly*, vol. 19, pp. 137–156, 1995.

[5] T. C. Powell and A. Dent-Micallef, "Information technology as competitive advantage: the role of human, business, and technology resources," *Strategic Management Journal*, vol. 18, no. 5, pp. 375–405, 1997.

[6] P. Weill and M. Broadbent, *Leveraging the New Infrastructure: How Market Leaders Capitalize on Information Technology*. Harvard Business School Press, 1998.

[7] W. Hasselbring, "Information system integration," *Communications of the ACM*, vol. 43, no. 6, pp. 33–38, June 2000.

[8] V. Sambamurthy, A. Bharadwaj, and V. Grover, "Shaping agility through digital options: Reconceptualizing the role of information technology in contemporary firms," *MIS Quarterly*, vol. 27, pp. 237–263, 2003.

[9] N. Melville, K. Kraemer, and V. Gurbaxani, "Review: Information technology and organizational performance: an integrative model of IT business value," *MIS Quarterly*, vol. 28, no. 2, pp. 283–322, June 2004.

[10] C. S. Chapman and L.-A. Kihn, "Information system integration, enabling control and performance," *Accounting, Organizations and Society*, vol. 34, no. 2, pp. 151–169, 2009.

[11] M. O. Land, E. Proper, M. Waage, J. Cloo, and C. Steghius, *Enterprise Architecture: Creating Value by Informed Governance*. Springer, 2009.

[12] R. K. Rainer and C. G. Cegielski, *Introduction to Information Systems: Supporting and Transforming Business*, 3rd ed. Wiley, 2011.

[13] R. Wagter, M. van den Berg, J. Luijpers, and M. van Steenbergen, *Dynamic Enterprise Architecture: How to Make It Work*. Wiley, 2005.

[14] R. Zarnekow, W. Brenner, and U. Pilgram, *Integrated Information Management: Applying Successful Industrial Concepts in IT*. Springer, 2006.

[15] W. M. Ulrich and P. H. Newcomb, *Information Systems Transformation: Architecture-Driven Modernization Case Studies*. Morgan Kaufmann OMG Press, 2010.

[16] J. Governor, D. Hinchcliffe, , and D. Nickull, *Web 2.0 Architectures*. O'Reilly, 2009.

[17] F. Xiaolong, L. Qixin, and Y. Fang, "Design and implementation of university level unified information system integration platform," in *Computer Sciences and Convergence Information Technology (ICCIT), 5th International Conference on*, 2010, pp. 864–868.

[18] J. Fong, *Information Systems Reengineering and Integration*, 2nd ed. Springer, 2006.

[19] J. Williamson, T. Laszewski, and P. Nauduri, *Oracle Information Integration, Migration, and Consolidation: The definitive guide to information integration and migration in a heterogeneous world*. PACKT Publishing, 2011.

[20] W. Ding and X. Lin, *Information Architecture: The Design and Integration of Information Spaces*. Morgan & Claypool Publishers, 2010.

[21] P. Bonnet, J.-M. Detavernier, and D. Vauquier, *Sustainable IT Architecture: The Progressive Way of Overhauling Information Systems with SOA*. Wiley, 2009.

[22] R. Richards, *Pro PHP XML and Web Services*. Apress, 2006.

[23] S. Abeysinghe, *RESTful PHP Web Services*. PACKT Publishing, 2008.

[24] L. J. Mitchell, *PHP Web Services*. O'Reilly, 2013.

[25] Information Systems Audit and Control Association (ISACA), "Control objectives for information and related technology (COBIT)," 2014. [Online]. Available: http://www.isaca.org/cobit