

## **Beyond the Sandbox: How Integration Researchers Can Actually Help Integration**

Arnon Rosenthal, Len Seligman, Barbara Blaustein  
{arnie, seligman, bblaustein} who work at MITRE.org

Large enterprises face daunting data integration challenges. Data integration researchers are busily turning out results, but they are often unready to be used at an enterprise scale. We believe that following a few simple principles may help and, incidentally, suggest interesting research questions.

Understand the enterprise: In earlier decades, most information management researchers began their careers solving enterprises' problems. Today's generation of researchers and developers are natives of the Web and merely immigrants to the enterprise. Relatively few researchers attempt to understand the concerns of enterprise managers, even as a thought experiment. Yet even an imperfect attempt to understand issues such as administrators' workload, incentives, and unreliability can pay dividends in design strategies that mesh well with enterprise processes. Researchers must begin to learn to use the levers that are available, such as administration budgets, differentiation among many roles (with consequent access privileges), and (partially enforceable) mandates.

Greater understanding of enterprise issues should lead to greater applicability to enterprise challenges. In [SR+], practitioners rated data cleaning as a critical problem, but researchers at the time did not. Furthermore, while researchers have claimed that schemas are undocumented (as they are, when scraped from the Web), we found that in DoD, ~90% of attributes and elements had textual documentation, averaging ~13 words -- enough for similarity comparisons. Cooperation is, of course, critical; it may often be infeasible to get data instances to outside developers, due to security and proprietary issues, because the target system was a service and did not store data, or because the target system was still under development [M+].

Allow for an imperfect world: Be explicit about how an idea works in a partly compliant system. Don't "assume a spherical sheep"; instead, create techniques that work if the sheep's shape is a sphere plus a deviation. Nearly every global assumption is violated somewhere in an enterprise. Don't start your research with a false premise -- instead, extend your algorithms to work over a mix of compliant and noncompliant parts. For example, if you invent a technique applicable to monotonic data mappings, describe how the general case can be simplified as a composition of monotonic and non-monotonic steps.

In some enterprises there may be multiple versions of a "fact". In many cases, provenance information must be able to be stored, tracked, queried, and annotated. Minimally, integration techniques must be robust enough to accommodate data versions and provenance; ideally integration techniques should exploit provenance information.

Find the right components: Research practice is an excellent paradigm in one important respect: it breaks problems into simpler pieces. It's important to have criteria for choosing the right decomposition.

*Divide by skill:* Costs balloon when each step requires both a domain expert and a programmer; narrower tasks are better suited to research, automation, and evolution. One helpful step would be to create freeware that popularizes reference models and suggested interfaces. In [M+] we proposed seven modular, single-skill steps for the schema mapping task.

*Separate tractable parts from residue:* Find useful and tractable "core" subproblems to solve, leaving well-defined specific "residue" problems for other solutions. For example, consider the theory of data exchange, which involves both schema mappings and constraints. The problem of deriving a data mapping from source to target schema is difficult when the target schema's constraints (e.g., null not allowed, key) require choices that are not implied by available metadata. Instead of giving up, split off a tractable problem: Create a schema whose structure matches the target, but whose constraints are just weak enough that mappings may be generated automatically. The residue problem (in this case, semantic choices between structurally identical schemas) is much easier for the user to understand. Meanwhile, researchers can expand upward from the empty set of constraints to identify other tractable cases.

Developers can use the case abstractions to compare and choose among alternate implementations, leading to gradual improvement.

*Choose core problems whose solutions can be fully automated:* Developers will be able to provide solutions in which some cases are handled completely, enabling more focused development of tools for the residue problems. System acquisition also reaps rewards: a well-defined automated capability can be included in proposal evaluation checklists. A final attraction is that an automated solution can be rerun “for free” to evaluate alternatives or when inputs change.

Work downstream: Other things being equal (dollars, delay, scarce skills), tools that are “downstream” -- whose results can be used directly in the operational system -- provide greater benefit. It is not an accident that vendors (unlike researchers) have spent most of their effort on generating executable transforms (“mapping”) rather than on identifying semantic relationships (“matching”). Downstream mapping improvements make some tasks “free”, e.g., changes to attribute representations that do not require new matching. “Free” results encourage new uses.

Play well with others: Both the database and the ontology community devote conferences to developing data integration technology, with little overlap [SE]. With some notable exceptions, the communities poorly understand each others’ concerns and strengths, e.g., standards and uniformity in the AI approach, and open world set semantics plus instant gratification [McD+] with databases. Few II researchers are engaged in helping the service-oriented computing community address mapping across tiers, clearly an information integration problem. We pay too little attention to data quality, security, and reputation management communities, or to database capability for tracking and annotating versions of a “fact”.

Researchers can be more effective by producing components that plug into frameworks. Such plugging-in been done for matching [DR], where all participants used roughly the same tasks (e.g., estimating match likelihood). We need to do it at larger scale. A community testbed might publish more precise descriptions of the tasks involved, including software interfaces. We also need to plug together larger systems, a challenge vendors face constantly as they acquire each other. Significant software engineering is involved, but the biggest challenge is (surprise!) data integration.

The situation is far from bleak. For the data security research community, we claimed the Giggle Test was desperately needed: Can you say, *without giggling*, that some rational manager might someday consider *your* idea the *best* payoff available for their labor and dollars? For data integration research, practitioners are not giggling. Many recent results from the data integration research community (e.g., theories of data exchange and entity resolution) have been both elegant and quite practical. Modest tweaks to current practices could improve the flow of ideas to practice and, consequently, the flow of practical problems (and funds) to research.

[DR] H. Do, E. Rahm. Coma: A system for flexible combination of schema matching approaches, VLDB, 2002

[McD+] L.McDowell et. al.: Mangrove: Enticing Ordinary People onto the Semantic Web via Instant Gratification. International Semantic Web Conference 2003

[M+] P. Mork, A. Rosenthal, L. Seligman, et. al., “Integration Workbench: Integrating Schema Integration Tools”, *Workshop on Database Interoperability at ICDE Conf.*, 2006

[SE] P. Shvaiko, J. Euzenat “A Survey of Schema-based Matching Approaches”, Journal on Data Semantics 2005

[SR+] L. Seligman, A. Rosenthal, P. Lehner, A. Smith, "Data Integration: Where Does the Time Go?", *Data Engineering*, Sept. 2002