Integration of Health Management Information Systems
Closed Standards vs. Open Source

Leif Arne Storset

8 January 2009
Abstract

Legacy systems remain common in software installations around the globe. This is especially observed in the field of health management information systems (HMIS) in developing countries. Legacy systems cost time and money and are expensive or technically difficult to upgrade.

Information infrastructure (II) theory proposes integration through gateways as a general strategy. The strategy shows promise, but developing robust solutions can be challenging. Extract-Transform-Load (ETL) tools offer more rapid gateway development and robust performance, and open-source solutions of this kind have recently appeared and gradually matured.

This thesis reports on an action-research project carried out with two HMISs in India. The Health Information Systems Programme (HISP) often needs integration between its open-source District Health Information Software (DHIS) and other HMISs. The leadership desired to explore whether open-source ETL tools could speed up development of gateways.

The integration project tried two approaches: first, the open-source ETL tool Pentaho Data Integration was utilized to create a gateway between DHIS 2 and a proprietary HMIS; then, an ad-hoc solution was created. The approaches were evaluated and compared.

In the course of development a number of interesting issues emerged surrounding the interaction between a proprietary legacy HMIS, the open-source HMIS and the open-source ETL tool. The limitations of legacy software that had been abandoned by its developers with no source code and little documentation available formed a striking contrast to the well-documented and actively developed open-source systems studied.

Furthermore, interesting social and technical dynamics of both forms of software surfaced: the developers of the legacy software had virtually disappeared from the picture, while the open-source ETL tool received continual enhancements during and after the project.
## Contents

1 Introduction 17  
1.1 Motivation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17  
1.2 The research project . . . . . . . . . . . . . . . . . . . . . . . 18  
1.3 Theoretical topics . . . . . . . . . . . . . . . . . . . . . . . . 18  
  1.3.1 Information infrastructures . . . . . . . . . . . . . . . 18  
  1.3.2 Fragmentation . . . . . . . . . . . . . . . . . . . . . . 18  
  1.3.3 Demands for integration . . . . . . . . . . . . . . . . 19  
  1.3.4 Organizational issues with integration . . . . . . . . . . 19  
  1.3.5 Legacy change . . . . . . . . . . . . . . . . . . . . . . 19  
1.4 Research objectives . . . . . . . . . . . . . . . . . . . . . . . 19  
  1.4.1 Expected research contributions . . . . . . . . . . . . 20  
1.5 Limitations . . . . . . . . . . . . . . . . . . . . . . . . . . . . 20  
1.6 Structure of the thesis . . . . . . . . . . . . . . . . . . . . . . 20

I Literature and Methods 23

2 Theory and Literature Review 27  
  2.1 Appreciating organizational impact on IS integration . . . 27  
    2.1.1 Stages of integration . . . . . . . . . . . . . . . . . . 29  
  2.2 Integration across knowledge boundaries . . . . . . . . . 30  
    2.2.1 Properties of boundaries . . . . . . . . . . . . . . . . 30  
    2.2.2 Levels of complexity . . . . . . . . . . . . . . . . . . 31  
    2.2.3 Solution mismatches . . . . . . . . . . . . . . . . . . 32  
  2.3 Health management information systems . . . . . . . . . . 33  
    2.3.1 Problems with current health information systems . . 33  
  2.4 Information infrastructures . . . . . . . . . . . . . . . . . . 34  
    2.4.1 Lock-in . . . . . . . . . . . . . . . . . . . . . . . . . . 36  
  2.5 Flexible standards . . . . . . . . . . . . . . . . . . . . . . . . 37
CONTENTS

2.5.1 AC/DC and network backbones .......................... 37
2.5.2 Flexible standards in HISP .............................. 37
2.5.3 An unorthodox view of gateways ....................... 38
2.6 Legacy systems ............................................. 39
2.7 Changing legacy systems .................................. 40
  2.7.1 Gaps between conception and reality ................. 41
2.8 Summary .................................................. 41

3 Methods ..................................................... 45
  3.1 Research methods ......................................... 45
    3.1.1 Participatory research ................................ 45
    3.1.2 Action research ....................................... 45
    3.1.3 Limitations ............................................ 47
  3.2 Case study ................................................. 48
  3.3 Research approach ........................................ 48
    3.3.1 Development and participation ....................... 48
    3.3.2 Observation .......................................... 48
    3.3.3 Meetings .............................................. 48
    3.3.4 E-mail ................................................ 48
    3.3.5 Documents and forms .................................. 49
  3.4 Summary .................................................. 49

II Empirical Study ............................................. 51

4 Context ....................................................... 55
  4.1 The Indian health care system ............................ 55
    4.1.1 Immunization services ................................. 56
    4.1.2 Health information reporting ......................... 56
  4.2 RIMS ..................................................... 57
  4.3 HISP ...................................................... 58
    4.3.1 Historical background ................................ 58
    4.3.2 Developer network .................................... 60
  4.4 DHIS ...................................................... 61
  4.5 Summary .................................................. 61

5 The integration project ..................................... 63
  5.1 Motivation ................................................. 63
  5.2 Settings .................................................. 64
    5.2.1 University of Oslo, Norway ......................... 64
### CONTENTS

5.2.2 HISP headquarters, Noida, India ......................... 64  
5.2.3 Ministry of Health, Gujarat, India ..................... 65  
5.3 Previous work ........................................... 65  
5.4 Databases and integration tools ............................ 65  
  5.4.1 Microsoft Access ..................................... 65  
  5.4.2 MySQL .................................................. 66  
  5.4.3 Pentaho Data Integration (formerly Kettle) ............ 66  
  5.4.4 Jackcess ............................................... 68  
  5.4.5 JasperReports and iReport ........................... 68  
5.5 Development process ...................................... 69  
5.6 Organizational issues ....................................... 69  
5.7 Summary .................................................. 70  

6 Analysis of standards ......................................... 73  
  6.1 UIP forms .................................................. 73  
    6.1.1 General statistics ................................... 76  
    6.1.2 Vaccination tables ................................... 76  
    6.1.3 Vaccine supplies ..................................... 77  
    6.1.4 Disease surveillance .................................. 77  
    6.1.5 Cold-chain equipment ................................ 77  
    6.1.6 Untoward reactions ................................... 78  
  6.2 Divergent philosophies: DHIS 2 vs. RIMS .................. 78  
    6.2.1 DHIS 2: Entity-Attribute-Value ..................... 78  
    6.2.2 RIMS: Snowflake ..................................... 80  
  6.3 RIMS schema issues ....................................... 81  
    6.3.1 Denormalized ......................................... 81  
    6.3.2 Unmarked foreign keys ............................... 83  
    6.3.3 Data types .......................................... 83  
  6.4 Reduction of Form 6 ...................................... 84  
  6.5 Summary .................................................. 84  

7 Pentaho gateway ............................................... 87  
  7.1 Creating a transformation ................................. 87  
    7.1.1 Extract ............................................... 89  
    7.1.2 Prepare data ......................................... 89  
    7.1.3 Look up additional data .............................. 89  
    7.1.4 Clean up data ........................................ 91  
    7.1.5 Load .................................................. 91  
  7.2 Differing “denormalization” terminology ................. 91  
  7.3 Loading data into Access .................................. 92
CONTENTS

7.3.1 ODBC .............................................. 92
7.3.2 Jackcess ........................................... 92
7.4 Data transfer using the RIMS user interface ............... 93
7.5 Source code and software access .......................... 94
7.6 Summary ........................................... 95

8 Ad hoc gateway ........................................ 97
8.1 Development process .................................... 97
8.2 Testing ............................................... 98
8.3 Cold-chain equipment .................................. 98
8.4 Other data issues ...................................... 98
8.5 Summary ........................................... 99

III Discussion and Conclusion .................................. 101

9 Organizational Integration .................................... 105
  9.1 The organizational domain ............................... 105
     9.1.1 Structure and culture .............................. 107
     9.1.2 Power and politics ................................ 107
     9.1.3 Strategy ......................................... 108
  9.2 Boundaries and novelty .................................. 110
     9.2.1 Boundaries ...................................... 110
  9.3 Evaluation ........................................... 111
  9.4 Synthesis ............................................ 111
  9.5 Summary ............................................ 113

10 DHIS and RIMS as infrastructures .......................... 115
  10.1 DHIS as an II ....................................... 115
      10.1.1 Flexible standards .............................. 116
  10.2 RIMS as part of an II ................................ 117
      10.2.1 Lock-in ....................................... 117
      10.2.2 Inflexible standard .............................. 117
  10.3 Legacy change through gateways ........................ 118
      10.3.1 RIMS as a legacy HIS ........................... 118
      10.3.2 Legacy change .................................. 118
      10.3.3 Closing gaps .................................... 119
  10.4 Gateway evaluation ................................... 120
      10.4.1 Pentaho ........................................ 120
      10.4.2 Ad-hoc solution .................................. 121
CONTENTS

10.5 Summary ......................................................... 121

11 Conclusion ...................................................... 125
   11.1 Contributions ............................................... 126
   11.2 Further work ............................................... 127

A Abbreviations .................................................. 129

Bibliography ....................................................... 131
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Different types of system inter-relationships ([Das92], quoted in [WW04])</td>
<td>29</td>
</tr>
<tr>
<td>2.2</td>
<td>[Car04]'s framework for boundaries between knowledge disciplines. The arrows indicate that novelty increases with difference between actors.</td>
<td>32</td>
</tr>
<tr>
<td>2.3</td>
<td>The Nordunet Plug. Illustration from [Han01]</td>
<td>38</td>
</tr>
<tr>
<td>3.1</td>
<td>The action research spiral, from [KM05, 564]</td>
<td>46</td>
</tr>
<tr>
<td>4.1</td>
<td>A timeline of HISP’s early history in Africa and India. Illustration from [BH02]</td>
<td>58</td>
</tr>
<tr>
<td>4.2</td>
<td>Hierarchy of standards where each level has freedom to define its own standards as long as they align with the standards at the level above. Illustration from [BH02]</td>
<td>59</td>
</tr>
<tr>
<td>5.1</td>
<td>A simple transformation as displayed in Spoon.</td>
<td>67</td>
</tr>
<tr>
<td>6.1</td>
<td>The form used by Primary Health Centres to report immunization statistics to districts, upper half. This part reports general information and vaccinations by vaccine type and demographics.</td>
<td>74</td>
</tr>
<tr>
<td>6.2</td>
<td>Lower half of the form in figure 6.1. This part shows vaccine and equipment supplies, reports disease observations and untoward reactions to vaccination, and equipment status.</td>
<td>75</td>
</tr>
<tr>
<td>6.3</td>
<td>A RIMS snowflake. This is only one of many possible snowflake diagrams one can create from the database.</td>
<td>81</td>
</tr>
<tr>
<td>7.1</td>
<td>The transformation intended to extract information on immunization sessions from DHIS 2 and load it into RIMS, as displayed in Spoon.</td>
<td>88</td>
</tr>
</tbody>
</table>
7.2 An illustration of comma-separated values. 

9.1 An illustration of this project’s meander through the various domains. It started in the power structures of the GOI (circled), or Government of India. We tried integration between the two systems, but ended with scrapping the legacy system and using DHIS 2.

9.2 Open-source software affected the technical, strategic and power domains in this project.

9.3 A comparison of [Car04] and [WW04], as applied to this project.
List of Tables

6.1  A comparison of DHIS 2 and RIMS database design. . . . . . 79

7.1  A simplified representation of sample data for the transfor-
     mation in figure 7.1. . . . . . . . . . . . . . . . . . . . . . . . . 90
Acknowledgments

I wish to thank my supervisors at the University of Oslo, Dr. Ola Hodne Titlestad and Dr. Jørn Braa. I wish to especially thank Dr. Sundeept Sahay for close supervision during my field work in India. Dr. Knut Staring, Lars Helge Øverland and master’s student Ole Kristian Hustad also extended invaluable help to me during the process.

I could not have completed my field work at HISP India without my always dependable and giving collaborators Bharath Kumar, Seid Hussein, Abyot Asalfew and John Lewis. Manvinder Kaur at the WHO in Delhi provided much assistance during the project, and her manager Rajeev Gera gave important information and contacts. At the Commissionerate of Health, Medical Services and Medical Education in Gandhinagar, Gujarat, India, Baghya Lakshmi was my partner and liaison during my entire stay there. I am grateful that she made Gujarat a second home for me. Mr. Kamesh Kumar Kantilal Panchal, Joint Director, D & E Cell, supported the project and gave advice.

Lastly, I wish to thank my family and friends, without whom I would not be where I am today. I cannot list everyone that helped me here, but I would like to mention Dr. Else Margrethe Berg, Douwe Beerda, and my parents. Thank you.
Trying new foods at the outdoor food court in Gandhinagar.
Chapter 1

Introduction

This thesis examines the topic of integration between closed-source legacy software and open-source solutions. The theories of information infrastructures, organizational integration and legacy change are applied. Research was conducted using the principles of action research, a qualitative method.

The empirical work was an integration project conducted in the cities of Delhi and Gandhinagar, India between January and May 2008. The aim of the project was integration of the legacy immunization health information system RIMS and the generic, open-source health information system DHIS 2.

1.1 Motivation

The Health Information Systems Programme (HISP) is concerned with offering lower-cost, more efficient health information systems (HISs) to developing countries. As in most other information systems (IS) contexts, legacy software — outdated software that cannot be updated — is a significant factor. To deal with it, the HISP project leadership wished to gain experience with open-source integration tools of a type called Extract-Transform-Load (ETL). Of such open-source ETL tools Pentaho Data Integration was judged to be the most mature candidate available in the fall of 2007. RIMS, a legacy HMIS used in many states in India provided an opportunity to both gain experience with open-source ETL and examine legacy integration from an information infrastructure viewpoint.
CHAPTER 1. INTRODUCTION

1.2 The research project

Research was conducted from January to May 2008, and followed the action research paradigm, which prescribes a cycle of planning, acting and observing, and finally reflecting, before repeating the process. The action research paradigm is explained more fully in section 3.1.2.

The goal was integration between the two health information systems DHIS 2 and RIMS, before eventually replacing RIMS with DHIS 2. There were three main phases to the work, constituting the planning and acting phases of action research:

1. Analysis of existing data standards, chapter 6
2. Attempted development of Pentaho gateway (i.e. integration software), chapter 7
3. Development of ad-hoc gateway, chapter 8

A gateway is a program that connects two separate systems. The intent was to compare a Pentaho with an ad-hoc gateway. Unfortunately, the Pentaho gateway could not be completed by the time field work ended.

After the integration project I reflected on and analyzed it, while the HISP India team completed integration and eventually replaced RIMS.

1.3 Theoretical topics

1.3.1 Information infrastructures

Information infrastructure theory deals with interconnected, heterogeneous information systems that behave in a sometimes counter-intuitive manner. The theory sheds light on how to exploit seemingly chaotic circumstances and generate value from heterogeneity.

1.3.2 Fragmentation

Foreign donors often require use of proprietary HISs to track progress toward development goals. In our case, the Universal Immunization Programme run jointly by the World Health Organization (WHO) and the Government of India required the use of RIMS, an application developed by a WHO contractor in Geneva. The result is a fragmented set of applications rather than a unified HIS. This is a common situation globally.
1.3.3 Demands for integration

The fragmentation of HISs and services leads to overlaps, gaps and a lack of standard definitions for data, reports and technological solutions. Fragmented HISs not only increase the burden to health workers at the peripheral level of the health sector but also increase the running costs and utilization of limited resources and limit the ability to obtain an overall picture of the health status of the community.

1.3.4 Organizational issues with integration

In recent years, social and political aspects of information system development have been emphasized, suggesting that integration should be not be considered simply a technical process. Other HISP authors have addressed a lack of empirical knowledge about the interplay of the political and technical configurations that arise during integration [SMA09].

1.3.5 Legacy change

Legacy systems, outdated software that cannot be updated, is very common. Such software is expensive to run, but also expensive to replace. The field of legacy change has developed strategies for legacy change that can help plan a switch from legacy systems. Developers may also use new principles of information systems design to reduce gaps between the rational plan and the chaotic reality.

1.4 Research objectives

Research objective Explore the impact of integration using open-source ETL software in a closed-source legacy setting.

This study investigated an integration project between DHIS 2 and RIMS. The traditional ad-hoc solution was attempted in parallel with a generic solution using the open-source business intelligence (BI) framework Pentaho, which incorporates an ETL component (formerly independent from Pentaho and known as Kettle). I evaluated the organizational impact of the legacy software and its replacement.

I will approach the research objective through these research questions:

RQ1 How can closed source and closed standards affect an organization’s integration process?
**RQ2** How can power, politics and culture in an organization impact the integration process?

**RQ3** Can user participation in information-system design reduce gaps between plan and reality?

**RQ4** Compare ETL software to ad-hoc solutions for integration.

### 1.4.1 Expected research contributions

This research is expected to contribute to strategies for building and integrating health information systems. The opportunities and limitations of open-source ETL tools and closed-source legacy software receives particular attention.

### 1.5 Limitations

The original intent of the case study was to implement both ETL and ad-hoc solutions for comparison. As the ETL solution was not successfully deployed in the field, we did not have the opportunity to compare it to the ad-hoc solution.

This work does not provide a detailed critique of the organizational frameworks used. In particular, they are not tested on further iterations of the action-research cycle. It would have been valuable to use them to attempt to predict and solve problems in real time rather than after the fact.

Although the research project has been participatory, additional participatory techniques, such as structured interviews and surveys, could have been applied.

The research has been conducted mainly by a single person, often isolated from collaborators. Larger teams would probably be able to explore the problems investigated in the project from additional technical angles.

### 1.6 Structure of the thesis

**Part I** provides an overview of the literature on health information systems, information infrastructure and other relevant topics. It explains the methodology used for this work.
1.6. STRUCTURE OF THE THESIS

Part II describes the empirical study conducted in India in partnership with the HISP team. It describes the context, the project framework and the phases of the integration work.

Part III elaborates on the findings from the field work and discusses them in light of the literature.
Part I

Literature and Methods
In this part, I will present the theories I built on during this project.

**Chapter 2** gives an overview of research on health information systems, organizational change and information infrastructures.

**Chapter 3** presents Action Research and my research approach.
My neighbors in Gandhinagar.
Chapter 2

Theory and Literature Review

This chapter provides an overview of the theoretical foundations of my empirical work. The case study is placed in the context of Health Information Systems research, especially concerning developing countries and organizational integration. A general introduction to the field of information infrastructures is given along with details on flexible standards. I will present research on legacy systems and current theory on changing them.

2.1 Appreciating organizational impact on IS integration

Other authors belonging to the Health Information Systems Programme (HISP) have addressed a lack of empirical knowledge about the interplay of the political and technical configurations that arise during integration [SMA09]. A theoretical framework for such knowledge is presented in [WW04], which identifies four domains of IS integration. The authors consider the first three mature and well-developed, while the fourth, the organizational domain, has not been explored in sufficient depth:

The technical domain is concerned with exchanging and translating information, putting “information at your fingertips”.

The systems domain contains the elements of a system and how they can be interfaced and integrated with each other, increasing coherency, coordination and control.

The strategic domain casts information systems in a central role in the competitiveness and success of an organization, so full integration
of business MISs and improving business processes is a priority.

The organizational domain

In a sentence, this domain involves “the integration of people, their ideas, and decision-making processes” [WW04, 336]. An integrated IS implementation may introduce new policies, processes and redundancies, and staff and management change power relations. A better understanding of such change is desired by managers (in a survey conducted in Britain by [DK01]), who recognize that “organisational issues were satisfactorily treated in less than a third of IS projects” [WW04, 336].

However, in the highly complex context of an integrated IS implementation, organizational issues are rarely treated up-front and proactively. Rather, the issues are analyzed reactively, after the integration has been attempted. This diverges from the approach used for the technical, systems and strategic domains, which are properly analyzed and whose changes are designed before implementation. The authors encourage using a similarly up-front approach to the organizational domain, and provide a strategic framework for analyzing organizational impact of the project. The proposed framework touches on the following aspects:

Structural changes may comprise “emergence of new functions, divisionalisation, de-layering, downsizing, strategic business units and inter-organisational strategies”, as well as business process re-engineering.

Social and historical context can be understood through developing a social theory together with the members of an organization. Such a theory should include historical “baggage” that can be important for explaining certain situations.

Power and politics can explain resistance to IS implementation, through resistance theories applied to people, systems (functionality and performance) and interaction between the system, its usage context and its political context.

Culture has a number of dimensions, including speed of innovation vs. aversion to risk, interdependence of colleagues, decision-making autonomy and management style.

Some hints on how to approach analysis of these subdomains is provided in [WW04].
2.1. APPRECIATING ORGANIZATIONAL IMPACT ON IS INTEGRATION

Figure 2.1: Different types of system inter-relationships ([Das92], quoted in [WW04])

2.1.1 Stages of integration

[Das92] defines terminology for four stages or types of integration (figure 2.1):

Stand-alone
- Elements make their own decisions
- No communication between elements

Interfaced
- Make decisions for own benefit
- One or two way communication

Integrated
- Make decisions for combined benefit
- Two way communication between elements
Universal

- No individual decision making
- Centralized control
- Single database

2.2 Integration across knowledge boundaries

Most innovation happens at the boundaries between disciplines or specializations [LB95]. This tells us that working across boundaries is a key ingredient of competitive advantage, but also why innovation proves so difficult to create and maintain. There is much theory on boundary management, and [Car04] has built a framework to help resolve incompatibilities between different perspectives.

2.2.1 Properties of boundaries

First, the authors identify three properties of boundaries, illustrating them with an empirical case from automobile manufacturer Beta Motors:

**Difference:** Actors have differences in amount and type of knowledge. For instance, Beta Motors has groups for safety, styling and engine design, each with specialized knowledge. This knowledge took investment in time and resources to acquire, and is “at stake”, meaning there are significant costs to giving it up and acquiring different knowledge.

**Dependence:** Decisions have effects across boundaries. At Beta Motors, a bigger, more efficient engine was developed. It raised the level of the hood, affecting the work of the styling group. Common knowledge must be developed to manage dependencies; this requires coordination.

**Novelty:** Previous common knowledge is insufficient due to a changing situation. For the automobile manufacturer, this could be new market demands for more “aerodynamic” styling, or the engine group’s new, improved design. Novelty would also be experienced for a newcomer not familiar with the common knowledge.
Novelty is often difficult to recognize. It is easy to think that the novel is something already known, because the thinking of a specialist is constrained to his or her specialty. This is a form of path dependence, past events having large impacts on future development [Han00, 64]. Such specialized thinking is an asset when actors have similar backgrounds and interests, or if they are independent — but a liability when they are different and dependent.

2.2.2 Levels of complexity

The framework orders three types of boundaries in rising level of complexity:

**Syntactic:** Under stable conditions, a common lexicon can be created and the syntactic boundary is unproblematic. But when novelty arises, the lexicon becomes outdated and insufficient, and groups no longer “speak the same language”. For Beta Motors, growing time-to-market pressures in the 1980s meant that the clay models used then were no longer sufficient for communication between various groups: their common language had broken down.

**Semantic:** This boundary is created when novelty makes some differences and dependencies unclear or some meanings ambiguous. Organizations can overcome this boundary by creating shared meaning and making tacit knowledge explicit. In the empirical case, the impact (or “meaning”) of engine design decisions on crash-test outcomes was difficult to agree on. New simulation tools developed by the company helped groups agree on importance of crash-test data and make more well-informed decisions.

**Pragmatic:** When interests are in conflict, the knowledge developed in one domain generates negative consequences in another. Novelty results in different interests among actors. Knowledge is invested in practice and so “at stake”. In the running example, the engine group had spent years developing a more efficient engine. But its larger size raised the hood of the car, going against current aesthetic trends that the styling group wanted to accommodate.

Carlile’s framework, modeled using the inverted triangle in Figure 2.2, recognizes that processes at the more complex boundary still requires the capacities of the less-complex boundaries. This means that shared interests require shared meaning, which requires a common lexicon.
2.2.3 Solution mismatches

The framework can describe mismatches that occur if a problem at one boundary is attempted solved at another, perhaps irrelevant, boundary. For instance, two teams may be facing a simply syntactic boundary — an insufficient clay model — but start more time-consuming work on developing shared meaning and interests.

Conversely, they may face semantic or pragmatic boundaries, and try to resolve them simply by using a common lexicon. This is more dangerous, because the unrecognized, unresolved novelty can have grave consequences. [Car04] illustrates this with a reference to [Chr97], who posits that good, innovative companies (such as hard-disk manufacturers) succeed and fail for the same reasons: because they listen to their customers. First, the hard-disk companies create disruptive hard-disk drive technology that enables new applications for customers. Then they develop more and more advanced technology, driven by their customers. Later, however, they miss strategically important opportunities to use less advanced technology, leaving the door open for startups to take their place.

The new modeling tool described in [Car04]’s case enabled communication across boundaries at all three levels, saving much downstream work. In the words of a climate-control engineer at Beta Motors, “We disagree sooner and know what we are disagreeing about more productively, since we have a shared way to compare our design parameters.”
2.3 Health management information systems

Good information is an absolute necessity when making management decisions — and there are few management decisions more important than those regarding health. A health information system is “an integrated effort to collect, process, report and use health information and knowledge to influence policy-making, programme action and research” [who00].

HISs, as information systems in general, may be categorized in three types:1

**Strategic** systems support high-level decision-making through simulation, financial forecasting and performance assessment.

**Tactical** systems are management information systems; more on that below.

**Operational** systems support day-to-day activities, for instance, electronic patient records, payroll, invoicing or purchasing/inventory.

We are concerned with the tactical type of system, the management information system, “a system that provides specific information support to the decision-making process at each level of an organization” ([Hur84] quoted in [LS00, 3]).

A health management information system (HMIS) is such a system. While it does not directly support the day-to-day responsibilities of a physician to cure people, it does gather information that ideally supports health decisions at every level, from the physician’s work, to monitoring preventive activities, to the long-term strategies of a country’s Ministry of Health or the World Health Organization (WHO).

The term health information system (HIS) is broader and may include operational or strategic systems.

2.3.1 Problems with current health information systems

Unfortunately, current HMISs are neither sufficient nor optimal to support solving the world’s health problems. Although the list in [LS00, 3-5] is almost ten years old, the problems mentioned there are still relevant today:

**Irrelevance of the information gathered:** While the HMIS could have been used to collect information to support management of health units

---

1These categories are taken from the University of Oslo course in Health Management Information Systems (INF5761), spring 2008.
and patients, data collection only partially supplies the needed information. Instead, data is skewed toward disease reporting.

**Poor quality of data:** Information collectors in peripheral health facilities frequently lack the technical skills, diagnostic equipment and training to gather the required data. Furthermore, feedback is only rarely given, leading to reduced motivation and incentive to ensure the quality of collected data.

**Duplication and waste among parallel health information systems:** Vertically structured “empires” founded with support from donor agencies create their own information systems. These tend to “focus on one one specific disease (e.g. diarrhoea), a specialized service (e.g. ‘family planning information systems’), or a management subsystem (e.g. ‘drug management information system’) instead of addressing management functions in a comprehensive way.” While quality of information was better, health workers were “drowned in a multitude of reports to be completed every month”, with no cross-references to prevent redundant data.

**Lack of timely reporting and feedback:** By the time data has been collected and prepared, it is frequently obsolete and no longer useful for decision-making, even for local managers who are closest to the people reporting it.

**Poor use of information:** Even data that is useful and relevant is not used, or its use has not been documented. This is especially a problem at the district level (below state level but above hospitals and health centers), despite efforts to encourage districts to use the information to make decisions — not just wait for decisions from central authorities. Ideally, districts would use “information for action”, not just “information for statistics”.

### 2.4 Information infrastructures

During the 1990s, information infrastructure theory grew out of the inability of information systems theory to explain and understand the complex reality of large, interconnected networks, the evolution and use of the Internet being both the prime example and the main theme of the field. Whereas information systems (IS) theory deals with constructing complete systems out of nothing or to replace existing systems, information infrastructures
2.4. INFORMATION INFRASTRUCTURES

(II) theory deals with open and evolving networks that already have an installed base. These networks are composed of individual information systems as well as users and other actors.

An information infrastructure may be summed up thus (based on a definition from [Han00]):

A shared, evolving and open heterogeneous installed base that is standardized in one way or another.

What follows is a brief overview of the fundamental terminology of II theory and how it applies to the archetype of such infrastructures, the Internet.

**Shared:** An infrastructure is shared in the sense that the same single entity is used by the entire community. For instance, the e-mail infrastructure we are used to today (SMTP) is the same for each of the more than one billion users of the Internet.

**Evolving and Open:** An infrastructure is never “finished” and has no finite “life cycle”. Rather, the users of the II collectively decide its evolution: they decide what features to use, support and develop, they select whom they communicate with and they choose when to stop using the II.

Furthermore, the II is open: new members of the infrastructure are continually accepted (although not necessarily indiscriminately), and there is no limit to the number of users, stakeholders and vendors involved.

The Internet has “always” been open to some category of new users: Arpanet accepted new military agencies, the Internet accepted new universities, and nowadays anyone who gains access to a terminal of some kind (computer, kiosk, mobile phone, etc.) can connect.

**Heterogeneous:** [Han00] lays out two main ways in which IIs are heterogeneous:

First, IIs are made not just of technological components: they also contain humans, organizations, institutions and so on. These non-technical elements are integral to the functioning of the infrastructure.

Secondly, “infrastructures are connected and interrelated, constituting ecologies of infrastructures”. An infrastructure is composed of ecosystems of sub-infrastructures by

- building one infrastructure as a layer on top of another
- linking logically related networks
• integrating independent components, making them interdependent

An example of layering is the “World Wide Web as a global infrastructure built on top of the Internet’s global TCP/IP infrastructure.” Logically related networks might be the transition of the Internet from IPv4 to IPv6. Finally, one might cite MIME as an integration of independent text, image, video and sound formats into a unified typology scheme for multimedia transfer.

**Installed base:** When is an II created? In this theory, never.

Infrastructures are always considered as existing already, they are never developed from scratch. When ‘designing’ a ‘new’ infrastructure, it will always be integrated into or replace part of an existing one. [Han00]

This applies to all transport infrastructures as well: every single road, even the first one, has been built in this way.

### 2.4.1 Lock-in

“For most of the history of computers, customers have been in a position where they could not avoid buying (more or less) all their equipment and software from the same vendor.” [Han00] Such a situation of stable equilibrium is generated by users investing in assets specific to a particular technology. It becomes very difficult, or sometimes impossible, to apply competing technologies. Switching costs (investing in the new system) are high, as are coordination costs (getting everybody to change at the same time). This can only be changed when other strong and significant influences come into effect which might push the system towards a new equilibrium [WR00].

An example of a lock-in is the aforementioned TCP/IP infrastructure used by the Internet, mentioned in [Han00]. The current protocol version, IPv4, has become outdated, and it was running out of available addresses at the end of the 1990s, and development of the new version, IPv6, started already in 1990. But the cost of coordinating a switch to the new version was high, and has not yet been completed. The effects of lock-in made compatibility with IPv4 the single most important requirement, leading the design of the new protocol to fulfill one goal only, increased address space.

The Internet example illustrates that to avoid lock-ins, the new solutions must support the transition from the old to the new. Section 2.5 introduces gateways, a concept that helps this transition.
2.5 Flexible standards

[BHH+07] introduces the “flexible standards” approach. This is a philosophy that rejects singular, universal standards as the only possible solution. Rather, it encourages modularization by defining multiple standards for a domain — each simple and limited in scope. The standards are tied together using gateways: people, processes or software that do the work of translating between data representations, formats, and protocols, and even media, like paper, digital storage and the spoken word.

Smaller and simpler standards are more change flexible, since it is easier to evolve them in response to changing circumstances. Rather than coordinating a change in a complex standard for the entire infrastructure, changes can be introduced in smaller standards for smaller sub-infrastructures.

2.5.1 AC/DC and network backbones

A familiar example is the little device on the power cord of every laptop computer: the rectifier (known as the power supply). This gateway allows equipment that requires direct current, such as your computer, to connect to the electrical infrastructure, which provides alternating current. Thus the two electrical standards coexist easily, albeit with some energy loss.

A classic example from the field of informational technology is the Nordunet Plug [Han01]. In the mid-1980s, universities in the Nordic countries worked together to establish a common, universal standard for communication. Several network communication protocols competed for attention, OSI’s X.25, IBM’s EARN, Digital’s DECnet, and ARPANET’s IP. Selecting the universal standard proved difficult, and the development of standards was taking years. As a pragmatic stopgap, the Nordunet Plug was proposed (figure 2.3). This was a network backbone that supported all four competing standards. With such an infrastructure in place, users started using the network, and services were maintained and expanded. Gradually, the “plug” gave way to an IP infrastructure as the Internet appeared and evolved.

2.5.2 Flexible standards in HISP

[BHH+07] presents the Health Information Systems Programme’s (HISP) DHIS software as an example of a system that supports flexible standards. The system allows each organizational level (from the World Health Organization down to the individual hospital) to define what information is
collected, allowing for differences between provinces and use areas.

Whereas South Africa successfully implemented a flexible standards strategy using DHIS [BH02], India is mentioned as an example of a well-established set of standards that are difficult to change. “It has proven very difficult to change the paper formats which are often ‘gazetted’ and appear to be cast in stone.” [BHH+07, 17]

### 2.5.3 An unorthodox view of gateways

In the traditional view of the computer communication field, gateways are seen as “an anomaly, something you need when you have failed to achieve a shared standard” [Ste94, BHH+07, 17]. The flexible standards philosophy views gateways as just as important as standards when building and maintaining flexible and successful infrastructures. They are seen as permanent members of the infrastructure, not just stopgap solutions. In the words of [BHH+07]:

Without gateways we are trapped. We need to develop infrastructures based on single, universal standards, but in countries like Ethiopia this is impossible. In most other cases, universal standards are beyond our capacity or will be totally inflexible if built. Establishing a fragmented infrastructure composed of a range of small ones which are not connected is not a viable option. An appropriate blend of standards and gateways allows infrastructures to evolve by maintaining order at the edge of chaos. [BHH+07, 18]
2.6 Legacy systems

Legacy systems, outdated software that cannot be updated, are one of the most challenging parts of information infrastructures. They are often created through lock-ins, and are usually kept around until the high switching costs are finally outweighed by time-consuming inconveniences or badly needed features that are impossible to add. When the change finally comes, users are often burdened with learning new procedures and changing long-established routines.

The challenges we experienced are strongly reminiscent of those encountered by [Sko03]. In Mozambique, legacy systems were characterized by nearly all the features mentioned in [BLWG99]:

- A lack of documentation means that tracing faults is costly and time-consuming
- A lack of clean interfaces makes integrating with other systems difficult.
- Extending the system is difficult, if not impossible.

Skobba also found the systems to be generally in a poor state. In particular, he characterized the HMIS used in Mozambique as a “black hole”: data was sucked in and never returned, and no-one could explain things that just seemed to happen [Sko03, 142].

Furthermore, and perhaps most ominously, “data stored in the existing systems were found to be redundant, inconsistent, non-uniform and ambiguous” [Sko03, abstract]. Such information structures can potentially have long-term ramifications:

Information itself—its structures in databases as well as the semantics of the individual data elements—is linked together into huge and complex networks that create lock-ins. One of the distinct features of information-based lock-in is that it proves to be so durable: equipment wears out, reducing switching costs, but specialized databases live on and grow, increasing lock-in over time. [SV99] (quoted in [Sko03])

Clearly, the need for change can be overwhelming for users of such systems. Legacy change theory, discussed next, deals with this sort of change.
2.7 Changing legacy systems

There is hope for organizations locked into legacy systems. After identifying the failings of legacy systems, [BLWG99] presents several methods for the transition from the legacy systems to the new system:

Cut and run: Switch off the legacy system and turn on a new, feature-rich replacement.

Phased interoperability: Replace a few of the existing components at a time.

Parallel operations: The legacy system and the new system are used in parallel. Once the new system is properly tested and trusted the legacy system is retired.

The authors consider approaches using gateways limited, because they do not support transaction management (meaning the data is not safe when being changed) and are difficult to build and operate [BLWG99, 7].

[Han00], on the other hand, strongly encourages gateways as a strategy. [Han00]’s terminology compares to [BLWG99]’s thus:

Revolution strategy resembles “cut and run”. Compelling performance may allow a quick switchover, or “flag day”, but the strategy is inherently risky and requires powerful allies.

Slow evolution uses backward compatibility and is an information infrastructure approximation of “phased interoperability”: certain locations install a new and more powerful system, but this system behaves like the legacy system as far as other locations are concerned.

Fast evolution uses gateways to link the new and the old networks.

Gateways (see section 2.5) can be especially important during the early phases of an infrastructure development. “There is still a considerable amount of uncertainty about how the infrastructure will evolve. And this uncertainty cannot be settled up front; it has to unfold gradually. Gateways may prevent those in the position of making decisions from acting like “blind giants” [Han00], making decisions without enough knowledge.
2.7.1 Gaps between conception and reality

[HMS99] provides a higher-level view of problems with HMIS design strategies. While mostly geared toward fresh implementations, it gives perspectives and opinions that are quite useful when upgrading to a new HMIS.

There is only limited understanding of what makes an HMIS succeed or fail, although “blueprints for success” abound. [HMS99] shuns such a blueprint attitude, and rather adopts a contingency approach to understanding HMIS successes and failures. This contingency approach looks at states of mismatch and match in implementation situations. The goal is to adapt to the various factors so that there is more match than mismatch.

This reluctance toward blueprints is quite compatible in philosophy with information infrastructure theory. With an evolving, open and heterogeneous infrastructure, it is reasonable to expect a variety of approaches to similar problems, and a variety of challenges in each case.

Striking the right balance is difficult: if the HMIS were to match the environment exactly, it would bring about none of the desired organizational change. On the other hand, if the HMIS tries to change too many of the surrounding factors, this brings with it a greater risk of failure. Overall, then, there is a trade-off between change and risk for HMISs.

In light of this, we may view integration using gateways as a reality-supporting strategy — the gateway helps the new system to match current realities.

[HMS99] suggests other strategies to minimize the gap between conception and reality. We will mention two of them here:

**Participatory design** includes users at every stage of the development process. This is “the bedrock of successful [HMIS] projects in a wide variety of settings” [HMS99, 19].

**Incrementalism** breaks a majors set of changes down and introduces them only slowly to reduce the extent of any given change. This has also been shown in practice to increase the likelihood of successful system introduction.

2.8 Summary

Research into organizational impacts of technical integration processes has suffered neglect, but [WW04] and [Car04] attempt to remedy this by providing frameworks for better understanding the domains and interests in-
involved. [WW04] breaks down the domains affected by integration processes, but more importantly further breaks down the organizational domain. This terminology can help more thoroughly prepare integration processes.

[Car04] looks at the issue from a different perspective: the integration of knowledge across various types of boundaries. It brings several theories of such boundaries together in a single framework, again providing a terminology for analyzing contingencies that arise.

Health information systems are immensely important for providing better care, but have a long history of suboptimal use. [HMS99] outlines typical problems found in HISs of the 1990s, many of which we will revisit in later chapters.

Information Infrastructure theory is helpful in understanding complex adaptive systems. Among other reasons, it is important because it views some “problems” — heterogeneity, change and legacy systems — as assets in an evolving infrastructures. Of course, these assets must be utilized well, and the theories of flexible standards and legacy change give pointers on how to change as smoothly as possible while making the most of existing resources.
Cricket with colleagues after a day of coding.
Chapter 3

Methods

Participatory action research formed the basis for this project’s methodology. Here, I describe this method and my specific approach.

3.1 Research methods

3.1.1 Participatory research

Participatory research is distinguished from conventional research in three attributes: shared ownership of research projects, community-based analysis of social problems, and an orientation toward community action [KM05]. If we interpret “social problems” narrowly as “gathering health information”, we see that much HISP research, including this project, easily falls into this category.

3.1.2 Action research

Participatory action research is iterative in nature (figure 3.1 on the following page), comprising several stages in each iteration [KM05, 563]:

1. Planning a change
2. Acting and observing the process and consequences of the change
3. Reflecting on these processes and consequences
4. Repeat

In this case, the planning was done by HISP leadership when they identified the requirement of more generic integration and the specific test case
Figure 3.1: The action research spiral, from [KM05, 564]
3.1. RESEARCH METHODS

of RIMS. My role was to act out the integration process and be participant observers, and to reflect on the process and results in this thesis. The first iteration also entailed planning, acting, observing and reflecting again, learning from my experiences.

For people used to traditional ideals of “hard” sciences such as physics, it is worth stressing that participatory action research investigates actual practices, not abstractions. “It involves learning about the real, material, concrete, and particular practices of particular people in particular places.” [KM05, 564] Of course, researchers can create and use abstractions, but “the here and now” is their principal concern, with all its history and complicated social relations and politics. They attempt to “avoid the delusions of the view that it is possible to find a safe haven in abstract propositions that construe but do not themselves constitute practice” [KM05, 564].

Like many other forms of qualitative research, participatory action research is critical. It aims to help people work toward freedom from constraints in their social media. Such constraints can be found in language, practices and power relations. These power relations may give experiences of affiliation and difference, inclusion and exclusion [KM05, 567].

In meetings with our customers at the UIP (section 5.6) they understood and encouraged the cycle of implementation – observation – improvement. They made the point that when solutions are not actually used in the field, momentum is lost. This happened to RIMS (section 4.2).

3.1.3 Limitations

A participatory project is by its nature bound to be influenced by the researcher. The researcher in such a project never pretends to be neutral, and it is a virtual certainty that another researcher would reach results that differ in some way from mine.

An action research project is not suitable for making generalizations in the same way as controlled experiments. It “does not aim to develop forms of theory that can stand above and beyond practice” [KM05, 568]. Conclusions from an action research project such as this may inform, criticize, develop and test theory — indeed, that is often a goal — but should themselves be interpreted critically.
3.2 Case study

Case study “is not a methodological choice but a choice of what is to be studied” [Sta05, 443]. This case is intended to shed light on the topic of integration of health information systems in the context of open-source and legacy software. As such it is an instrumental case study, and I am interested both in the particular and the general. [Eri00]

3.3 Research approach

A variety of action research techniques was employed, both more traditional ones like observation and techniques particular to action research, such as development and participation.

3.3.1 Development and participation

The HISP project is coordinated from the University of Oslo, with development teams in several countries, among them India. This team works autonomously with Indian customers, representing HISP during the entirety of a project. (See section 5.2.2 for more on this team.) I participated in day-to-day work at its offices near Delhi. I primarily worked on the integration project but helped out in other tasks as well, both technical and in project management. I kept detailed notes from my work.

3.3.2 Observation

I was allowed to observe HISP meetings with customers. This gave insight into organizational issues and the mindset of the various parties. I also conducted user tests.

3.3.3 Meetings

Meetings with the customer, both formal and informal, helped me better understand their requirements and concerns. Developer meetings allowed me to keep track of DHIS 2 development in general.

3.3.4 E-mail

Most global communication in HISP was done using e-mail, and much of this communication used mailing lists. By observing discussions on the
mailing lists I could pick up developments that concerned the integration project and HISP in general. Regular e-mail helped me keep in touch with program coordinators in Oslo.

3.3.5 Documents and forms

Analysis of Form 6, Form 9 and UIP forms was the foundation for chapter 6. I also examined the RIMS user manual, although I quickly found it to be of limited value to us.

3.4 Summary

Choosing participatory action research as the research method lets us as researchers transform both theory and practice. It shows the results of change and examines the concrete experiences of real people, and is critical toward existing practice. However, it is not “hard” science and cannot by itself be used to generate universal “laws”. This project involved a variety of qualitative research techniques, including observation and participation.
Part II

Empirical Study
This part provides details on the integration project and its context.

**Chapter 4** gives an overview of the political, and organizational context of the integration project.

**Chapter 5** describes the project, including software used, the development process, and anticipated organizational issues.

**Chapter 6** analyzes existing data standards.

**Chapter 7** recounts development of the ETL gateway.

**Chapter 8** summarizes development of the ad-hoc gateway.
Children of Ahmedabad.
Chapter 4

Context

This chapter describes the context of the integration project. The Indian health care system is summarized along with details on health information reporting. Then I introduce the various players and software involved in the project.

4.1 The Indian health care system

An integrated and comprehensive health care system has long been the goal of government policy in India. The National Health Policy adopted in 1983 focused on this, and since then an extensive health-care system covering the entire population has spread to over 600,000 villages. However, many key health indicators are low, communicable diseases continue to be a major problem, maternal mortality is high, and morbidity, especially among the poor, exacts a high toll [Joh05, 57].

The private sector of the health-care system is larger in volume than the government sector. The sectors are described as follows in [Joh05, 57]:

The government sector provides publicly financed and managed curative and preventive health services from the primary to the tertiary level, throughout the country and free of cost to the consumer. These account for about 18 % of the overall health spending and 0.9 % of the GDP.

The private sector plays a dominant role in the provision of individual curative care through ambulatory services and accounts for about 82 % of overall health expenditure and 4.2 % of the GDP.
There is a tradition for cooperation between the sectors in “public-private partnerships”, where governments co-operate with private enterprises to provide services that are better or cost patients less. For example, the Government of Gujarat pays private practitioners to provide free obstetric care to poor women. This private-public partnership, called “Chiranjeevi Yojana”, or “plan for a long life”, was launched in 2005 in five districts, and now covers all 25 districts of Gujarat [Wor09].

At the top government level, administration is led by a Secretary for Health and a Secretary for Family Welfare. The rest of the organization is mostly program/project based, creating ad hoc organizations for projects such as tuberculosis or malaria. Many states have separate structures for family welfare operations since population control through family planning is given great importance [Joh05, 57-58].

States usually have separate directorates or head offices for primary, secondary and tertiary health care, and they also direct medical education. The state of Gujarat, where part of the field research for this project took place, had head offices for health services in Gandhinagar, the capital city. For the most part we worked with the staff at the Commissionerate of Health, Medical Services and Medical Education, under the Joint Director.

4.1.1 Immunization services

As mentioned above, a large part of government health services are organized in vertical programs. This also applies to immunization, which is administrated by the Universal Immunization Programme. The UIP is supported by UNICEF and other donors.

The state health system is divided into districts. In each district there are several Primary Health Centres (PHCs), each with several Subcentres (SCs). The SC has an Auxiliary Nurse Midwife (ANM) who travels around the community and carries out immunization and other health services, including, obviously, midwifery duties.

4.1.2 Health information reporting

The ANM reports immunizations performed to the PHC and the district using a standard UIP paper form called Form 6. The district may then enter the data into the Health Information System. In Gujarat, DHIS 2 is used at both the district level and at the Ministry of Health.

The ANM also reports the same data (plus some information on vaccine stocks) to the UIP. This illustrates a disadvantage with vertical programs
4.2. **RIMS**

such as the UIP: they create their own structures and hierarchies in parallel with other health programs, as well as walled-in information systems. This historical “baggage” leads to duplication in both collection and reporting (see sections 2.1 and 2.3.1):

**Collection:** There is a risk that multiple programs will collect the same information from the same people, leading to wasted time and money, and ultimately fewer services to the population.

**Reporting:** Similar forms are filled in manually twice, which increases the risk of error as well as wasting time and money. Inaccurate, inconsistent and missing data can lead to differing views of the same situation by different actors.

4.2 **RIMS**

The Routine Immunisation Monitoring Software (RIMS) is a tool for reporting immunization of infants and pregnant women through the Universal Immunisation Programme. SOL, a WHO contractor in Geneva, was hired to develop RIMS and train users during a three-month program. Two versions of RIMS were developed by them in 2004:

- An offline Visual Basic application using a Microsoft Access database, used in districts.
- An online .NET application using a SQL Server database and a World Wide Web front-end.

In Gujarat, offline versions are installed in the district offices, where data entry is performed before exporting data to an XML format and uploading to the online version, which is hosted at state offices.

Some of the information reported through RIMS is also recorded in other Indian reports, such as Form 6 and Form 9, which leads to redundant data entry and duplicate efforts by employees of Primary Health Centres, blocks or districts — wherever the information is entered. The integration project aims to remove these redundancies, and also yield synergies in reporting: with all the data available in one place, it is easier to see the big picture.

Unfortunately, the state was only informed of development, not involved, so they were given no ownership to the project. In January 2008 only 50 districts were reporting using the system. SOL was rehired that spring to fix defects and deliver the source code.
4.3 HISP

The Health Information Systems Programme is concerned with offering lower-cost, more efficient health information systems (HISs) to developing countries. It consists of a network of developers working to make HMISs available and affordable to governments in the developing world. The main product, the District Health Information Software (DHIS), is in development or use in 15 countries around the world. There are developers in India, Vietnam, Ethiopia, Tanzania, Tajikistan, Ireland, and Norway [dhi09].

4.3.1 Historical background

As described in [BH02], HISP had its beginnings in the fall of apartheid and advent of democracy in South Africa in 1994 (see timeline in figure 4.1). At the time, 60% of resources allocated to health care served 20% of the population, the white minority. The African National Congress, the ruling party after the first democratic elections, launched the National Health Plan (a part of the larger Reconstruction and Development Program). A decentralized system of health districts was envisioned, along with a new national health information system to support the restructuring of the health sector.
In 1996, the HISP project was founded by researchers at the University of Oslo, staff from two Cape Town universities, activists from the health sector and NGOs.

Development of the DHIS started in the first project phase (1996–1998), during which new national health information standards were developed. This bottom-up process resulted in a hierarchy of standards, illustrated in figure 4.2: a core set of data elements used nationwide, and supersets of data elements determined by the individual districts.

The lower levels of the health services will generally need more detailed data—that is, larger data sets or longer 'lists,'—in order to support their day-to-day work than will the higher levels, in order to support their coordination and overall management.

[...]

A process of prototyping, negotiation, and tinkering to develop a national standard data set acceptable for all provinces and national health programs. All provinces have now incorporated these national standards into their provincial data sets. [BH02]

This hierarchy of standards ties in with the Flexible Standards philosophy (see section 2.5): a minimal data set that was compatible with a multitude of other standards.
The action research paradigm, used widely in HISP since the very beginning, has caused HISP to always explicitly and implicitly see themselves as political actors in a larger development process.

As an independent open-source project working closely with governments around the developing world, HISP has a complex relationship with existing power structures as well as organizational reform. [BH02] summarizes a basic ambivalence in the HISP process thus:

HISP has never been fully accepted by official structures, but never fully dismissed neither. Being a bottom-up, grass-roots movement, this is probably inevitable and indicates both the strength and the weakness of a movement like HISP: independence and vulnerability. [BH02, 119]

In reforming health reporting organizations, the organization would collide with an unexpected psychology of existing practices. The existing data reporting system had not only the intended purpose of reporting on activities in the injection room, but also the unintended consequence of confirming and reinforcing social contracts and existing power structures: viz. a finished report as “proof” of work and new reporting duties as “punishment from above”.

In South Africa, the team’s working methods surprised users. Participatory prototyping astonished users accustomed to “drawn-out tender processes, fully pre-specified development projects that often ended in frustrating delays or fiascos.” The project followed a meritocratic approach to user involvement: any user, regardless of his or her place in the hierarchy, had full access to the development team. But, as the team admits, “Such guided user participation is obviously time-consuming and only possible with a limited number of users.”

4.3.2 Developer network

Formerly, project leadership and most development work was carried out in and from Oslo. In recent years, however, the project has been transitioning to a “South-South-North” mode of operation. Nowadays HISP organizations in other countries, especially India, operate more independently, though in close collaboration with the University of Oslo. DHIS 2 is undergoing heavy development in India, with a small team of developers situated in Delhi and implementers in several other offices, notably Kerala.

The class INF5750 “Open Source Software development” is offered every semester at the University of Oslo. DHIS is used as an example system,
and students are assigned projects contributing to the further development of the HMIS. The class is compulsory for many students, and it serves dual roles as an introduction to open-source software development and a recruiting ground for potential project members.

There is some degree of collaboration with organizations behind other HISs such as OpenMRS, which is an electronic patient record system.

4.4 DHIS

As explained in section 4.3.1, DHIS was conceived in an atmosphere of empowerment and grassroots activism. This is reflected in the fundamental flexibility built right in. It is exploited along three dimensions. First, the total freedom of customization affords DHIS 2 great change flexibility. Requirements do change over time, even from a well-established actor such as the WHO, and when they do, the system can be updated without the expense and logistics of involving professional programmers.

Second, data elements can vary between organization levels. A PHC may choose to monitor very specific health problems unique to its population, whereas the Government of India has a narrower range of indicators that apply to the whole country. The same applies to districts and state governments. A flexible approach accommodates all levels at the same time.

Third, and perhaps more subtly, data elements are no longer constrained by their respective paper forms. Certain data may be collected using multiple forms for the same population, but they are treated as the same data. If the same data in different forms were treated as two different data elements, one would have duplication of data.

4.5 Summary

India has a dual health-care system, with a large private sector and a smaller public sector that often cooperate. The system has been continuously expanded since 1983, but significant health problems remain. Immunization is administrated through the Universal Immunization Programme (UIP), a vertical structure with its own reporting structure that leads to duplicate reporting. RIMS was the custom software used to collect information for this program. HISP is a non-profit organization promoting DHIS 2, an open-source health information system. Together with the Indian health sector the organization initiated a project, described next, to reduce redundancy in health information reporting.
The Ministry of Health in Gandhinagar, Gujarat.
Chapter 5

The integration project

This chapter explains the motivation for the integration project. It presents the main settings of the research and previous work on the subject, and gives an overview of software tools used, along with some technical background. An overview of the development process and some presumed organizational issues is given.

5.1 Motivation

As in most other information systems (IS) contexts, legacy software is a significant factor in HISP, and it must be dealt with. As explained in section 2.7 and [BLWG99], there are several ways of approaching legacy systems: replacement, coexistence (and perhaps duplicating features and work) or integration. Each type of solution has been used at various times in HISP’s two-decade-long history, and in this project, integration was seen as a stepping-stone to replacement.

In HISP, integration is usually done on an ad-hoc basis with custom integration modules written for every project. However, program managers desire a more generic strategy for integration, where ETL software (such as that described in section 5.4.3) is used to make integration simpler, quicker, more robust and more reusable, as well as easier to understand and modify when changes are required.

HISP has a strong tradition for using and promoting open-source software. Therefore, project leaders wanted experience with open-source integration tools. Integration with RIMS was chosen as a test case to learn about the field, gain experience and test the available solutions and infrastructures.
This integration project was only one out of several that HISP initiated to this end. In the semester when this project was initiated, the University of Oslo class INF5750 (see section 4.3.2) had six different projects related to Pentaho Data Integration (called Kettle at the time), including a “dashboard solution” that presented health information graphically and allowed breaking down the numbers visually [TØ07]. I participated in a project working to aggregate data from both DHIS 2 and DHIS 1 at the same time, even though they ran on quite different database systems. (For more information on the database systems, see sections 5.4.1 and 5.4.2.) The master’s project presented in this thesis can be seen as a continuation in spirit of the class project.

In the terms of [Das92] (section 2.1.1), this integration project is actually an interfacing project, not an integration project, since it was for the benefit of one of the systems and the other was to be replaced.

5.2 Settings

5.2.1 University of Oslo, Norway

As described in section 4.3.1 on page 58 the HISP project was founded, among others, by researchers at the University of Oslo. The project is still to a large degree coordinated by University of Oslo staff, although efforts are underway to make project leadership even more widely distributed (section 4.3.2).

I learned to know about HISP through a class on open-source software development taught by Ola Hodne Titlestad, Lars Helge Øverland, Jørn Braa, Ole Hanseth and others, and was invited to do my master’s project with them. The class introduced me to DHIS 2 development, and I kept in touch with many project members throughout the work on this thesis in addition to my advisers.

5.2.2 HISP headquarters, Noida, India

The main part of development, sales and project management on DHIS 2 for use in India is done by HISP staff in the National Capital Region of India. HISP development headquarters at the time were located in Noida, a few minutes’ drive from New Delhi. (The development team has since relocated to New Delhi.) Development was mainly carried out by a team of five software engineers and a project leader. During my two and a half
months in Noida I worked closely with the team to prepare and perform integration with RIMS.

5.2.3 Ministry of Health, Gujarat, India

Final development and testing of the integration solution was performed at the Gujarat Ministry of Health headquarters in Gandhinagar. I spent one month there working closely with Ministry staff and fellow HISP developers. A user test was performed at the Gandhinagar health district office, which was located near the Ministry.

5.3 Previous work

Ole Kristian Hustad had made an earlier attempt to import data from RIMS to an Excel reporting format [Ole08]. His work was hampered by the lack of access to database schemata, since he never received the required password (which I received after some time).

Some of his analyses of missing data in DHIS 2 were of use to me, although our export requirements were too different for his code to be useful.

5.4 Databases and integration tools

On the technical side, several pieces of software were used in the integration process. Pentaho Data Integration and JasperReports were used in the attempt to integrate DHIS and RIMS. The two systems ran on different database management systems, DHIS on MySQL and RIMS on Microsoft Access. (On server installations, RIMS could run on SQL Server, but I was not allowed to use this version of the system.)

Microsoft Access is the only closed-source, proprietary software used in the project. Its competitor MySQL was acquired by Sun Microsystems during the integration project, and Sun was acquired later by Oracle Software, but the source code remains open.

5.4.1 Microsoft Access

Microsoft Access, part of the Microsoft Office suite of applications, is a desktop database management system that allows programmers to quickly create applications backed by databases. As such it is suitable for rapid prototyping of new software, or smaller ad-hoc databases.
The platform was used for DHIS 1 from the late 1990s, and DHIS 1 also made use of Access’s software development tools in addition to storing data in the database. DHIS 2 uses full-featured SQL databases instead.

Access is not primarily intended as an enterprise database management system, and it has shortcomings when used as such. For instance, it has rather incomplete support for SQL, the standard language for communicating with databases. Furthermore, the database file format is secret [Mil08], so accessing the files requires using Windows, or else reverse engineering the format. With both options – the popular SQL, and reading the database file directly – limited or closed, there were fewer options available to us in the integration project.

5.4.2 MySQL

MySQL is a full-featured database management system, developed by MySQL AB. It is dual-licensed, meaning that users may choose between the GNU General Public License, which costs nothing but requires the software to remain free, and a proprietary license, which allows making secret changes and putting it into proprietary software but costs money.

MySQL is a popular option for storing the DHIS 2 database. While DHIS 2 is database-independent and directly supports systems such as PostGreSQL, MySQL’s market share means that many go for this option.

MySQL has good support for SQL and advanced database features that are useful in an enterprise environment with many users and heavy traffic load. Significantly, it is well documented and well integrated with many programming languages and integration tools, such as Pentaho Data Integration.

5.4.3 Pentaho Data Integration (formerly Kettle)

Pentaho Data Integration (PDI) is an open-source extract-transform-load (ETL) framework. This means that it is able to extract data from one source (such as a Microsoft Access database), transform it so it fits a target (such as a MySQL database), and load it there.

At the time of this project PDI was leading the pack of open-source ETL tools, so HISP leadership chose to explore it as part of an integration strategy. This integration project was one of several that HISP initiated to this end.

It is designed to be enterprise-ready and compatible with a wide range of systems, while also being easy to use and well-documented. The solu-
5.4. DATABASES AND INTEGRATION TOOLS

Figure 5.1: A simple transformation as displayed in Spoon.

In PDI, the ETL process is designed using the following hierarchy, from the smallest to the most overreaching:

**Step:** A single action, such as extracting data from or loading to a database, adding values together, or more advanced modifications such as (de)normalizing database rows (see below).

**Transformation:** Any number of steps may be combined into a transformation, which is essentially a small computer program (a “script”). Instead of writing program statements as one would in other scripting languages, the user may use Spoon to draw little boxes and lines on the screen representing program flow (see figure 5.1). For instance, the user may begin with a step that reads data from a Microsoft Access database, draw a line to a step that normalizes the data, and then connect these steps to a final step that outputs data to a MySQL database.

**Job:** A job allows higher-level control of transformations. For instance, a job may be designed to read a configuration file and a database and decide which transformation to run, with which parameters.

These are some of the terms used in PDI steps:

**Input and Output** refers to extracting data from a source and loading it into the target.

**Normalization** refers to reducing redundancy within a database schema. An example from the RIMS database: there is a table, `district_mst`,
that assigns a code, \texttt{district\_code}, to every district and shows which state it belongs to. But in many other tables — indeed, every table that uses the \texttt{district\_code} — the state code is repeated along with the district code. This redundancy makes it very difficult to maintain consistency within the database; for instance, one might accidentally assign a district to different states in different tables. Normalization is the process of removing such redundancy; denormalization would be to introduce redundancy. The latter is useful when exporting data to a denormalized system such as RIMS. See section 6.3.1 for more information on the RIMS schema.

5.4.4 Jackcess

Jackcess is a library (that is, a collection of program snippets) that enables working with Microsoft Access database files using the Java programming language. No Microsoft software is required, and it is therefore built into Pentaho Data Integration (which is free software).

Jackcess is part of a family of open-source projects run by Health Market Science. The company services top pharmaceutical and device manufacturers and pharmacy chains. The other projects are also libraries and tools, and they are developed to support Health Market Science’s commercial health management information systems.

Since the Microsoft Access database-file format is secret, the Jackcess project must reverse-engineer it in order to read and manipulate such files. Consequently, Jackcess is not as reliable as Access when working with these files.

5.4.5 JasperReports and iReport

For creating reports from Java programs, JasperReports is a popular choice in the open-source world. It allows programs to generate reports from a variety of data sources, such as Microsoft Excel or SQL databases, in a wide selection of formats, such as PDF or files that may be viewed on the Internet (HTML) or in Microsoft Word (RTF). iReport is a tool used for the visual design of the reports.

HISP uses JasperReports for reporting in several cases, and I attempted to use it in this project as well.
5.5 Development process

We explored several options for exchanging data between RIMS and DHIS. The preferred option was an ETL framework such as Pentaho. Most of my effort was spent following this path. The advantage of an ETL framework is that future modification would require modifying only metadata, not program code. This would lead to a much more robust solution.

One requirement (at least in the first phase of implementation) was outputting the immunization information on the same form format as the UIP had used before. For this I explored JasperReports, which had been used in HISP previously. I used the GUI, iReport, to attempt to design reports that mimicked the forms that had been used. Unfortunately, the type of complexity of the UIP form was not a good match for the features of JasperReports. The software is geared toward reports of data can be represented in a few fact tables, not the amalgamation of many data sources such as in the UIP forms.

Early on, Pentaho Data Integration was explored as an option for transforming data for export. PDI showed much promise, with simple tools to modify transformations. However, the Microsoft Access component of PDI was not able to cope with large, production-size databases, and other methods of connecting to the database failed, perhaps because of errors in PDI. Errors in RIMS further complicated matters. The promised source code for RIMS never materialized, so the errors could not be fixed.

For this reason, a custom DHIS module was developed to run the export and import. While this meant that data transformations have to be written with more difficulty and are not as clear and modifiable, there are advantages to this approach: error logging is easier, and integration between DHIS and PDI is not an issue. In other words, we only needed to integrate the two, and not PDI as a third component.

5.6 Organizational issues

In meetings with UIP and HISP representatives, these representatives expressed views on organizational change. They had experienced that such change is slowed by organizational inertia and resistance. One strategy to prevent this is to change the organization gradually, and only after introducing the new software. Although effects of rationalization are not as immediate, no one need feel threatened by new technology.

Allowing the organization to feel ownership and need of the project
is important for it to retain the necessary support. The best software in the
world will not do any good without “high protectors” that understand how
it can help, and has helped before.

For field implementation, they recommended using competent people
already out there, rather than bringing new people, and identify key per-
sonnel in each district.

They also identified cultural problems which they had encountered in
the past. Specifically, organizations typically use information for political
purposes (though this applies to any kind of organization anywhere, not
just health organizations). For instance, if the documented immunization
rate in a district drops from 85 % to 60 % because of better-quality data,
someone will ask why, and top management may find the figures difficult
to defend. In extreme cases, a middle manager may work with the 60 %
figure, but report the 85 % figure to politicians.

5.7 Summary

HISP and its customers in the Indian public health sector wish to imple-
ment integrated health information reporting using DHIS 2 to reduce re-
dundancy. HISP also wished to use this case to explore use of open-source
ETL integration tools to speed up future integration projects.

This chapter provided a broad overview of tools used in the integration
project and in HISP generally. It covered database management systems,
integration tools and libraries, and collaboration tools.

Organizational issues were anticipated, although unexpected problems
of both organizational and technical nature interrupted and redirected the
project at several points. The generic ETL solution was abandoned, and an
ad-hoc approach was tried instead.
5.7. SUMMARY
Health administrators at Gandhinagar district offices. (Photo by Sunil)
Chapter 6

Analysis of standards

Before I could work with extracting, transforming and loading data between the various systems, I had to analyze the existing data standards — that is to say the database schemata of DHIS 2 and RIMS. This chapter describes the results of the analysis, both as concerns the database schemata and the actual health information collected in them. Also, differences between the DHIS 2 and the RIMS schemata are described, and RIMS’s schema in particular is critiqued. Finally, an effort to reform existing Indian data standards is described.

6.1 UIP forms

I was given a RIMS user manual, but it did not explain the database tables or their purpose. Because of this lack of documentation I had to start analyzing the database schema on my own, consulting with experts at the Universal Immunisation Programme (UIP) when necessary.

I will begin by presenting the health information collected by the UIP. The monthly UIP forms (see figures 6.1 and 6.2) require data of many different kinds to be combined, as presented in the following subsections.

The following mainly describes the PHC version of the form. There are also district and state versions; these contain financial and supervision fields in addition to the fields presented here. On the district form, the other fields (for immunization, disease surveillance and equipment inventory) are summarized from the PHC data and reported to the state immunization director; similarly, district data is summarized and reported to the national level immunization coordinators using the state form.
### Figure 6.1: The form used by Primary Health Centres to report immunization statistics to districts, upper half. This part reports general information and vaccinations by vaccine type and demographics.

![Image of the form](image_url)
Figure 6.2: Lower half of the form in figure 6.1. This part shows vaccine and equipment supplies, reports disease observations and untoward reactions to vaccination, and equipment status.
PHCs fill in the forms on paper and hand them to the district offices, where the data is entered.

Note that there is much duplication of data in the form, which has many fields cumulating year-to-date figures. This means that the PHC reporter must consult the health center’s own statistics for the previous months. This extra work may have been designed to increase quality control, but also increases the number of places where error may be introduced.

### 6.1.1 General statistics

The first part of the form gives the big picture of the PHC’s immunization efforts. The following information is gathered:

**Yearly targets** for infants and pregnant women are reported every month. These are based on population figures. The number of fully immunized infants — those that have received all necessary vaccines — indicate progress toward the target.

**Number of sessions** planned and held. Some sessions are held at Aanganwadi centers, which are village centers for basic health services.

**Volunteers** are needed to mobilize children since one cannot summon everyone by mail.

**Private vaccinators** are contracted where there is no Ambulatory Nurse Midwife (ANM), where there is not enough health personnel to serve an area, or in urban slums.

**Supervisory visits** (district and state forms only) are undertaken by District and State Immunization Officers to inspect progress.

### 6.1.2 Vaccination tables

A wide variety of illnesses are prevented using the program’s vaccines:

**Tetanus toxoid** one of several tetanus vaccines

**BCG** tuberculosis

**OPV** polio

**DPT** diphtheria, pertussis and tetanus

**DT** diphtheria and pertussis
Hepatitis B

Measles

In addition, Vitamin A supplements are given to boost the immune system.

The vaccines are administered in 1–5 doses, some with an addition “booster” vaccine (denoted “B”). The immunization data is structured along several dimensions:

- vaccine
- dose (also called “antigen”)
- age range (under/over 1 year, pregnant women)
- gender

Gray fields indicate immunizations that do not occur; for instance, the 5-year DT vaccine is not given to infants under 1 year of age.

6.1.3 Vaccine supplies

For each vaccine and syringe type, this table shows how many vaccines were received, used and found unusable, and how many were left at the end of the month.

6.1.4 Disease surveillance

This table tracks how many children have contracted or died from a disease.

6.1.5 Cold-chain equipment

Some vaccines contain live organisms that may become dangerous if allowed to reproduce. The cold chain is designed to keep vaccines cold from manufacture through storage, transportation, distribution and administration, even in hot climates.

This section is different from the rest of the form in that it is not statistical. Instead of sums totaling the work done on a population, it tracks the inventory of individual pieces of equipment needed to maintain the vaccine cold chain, such as freezers and refrigerators. The equipment is tracked by make and serial number and state and progress of repairs.
This section proved difficult to represent in DHIS 2, since the software is designed to be used for purely statistical data, not inventory. See section 8.3.

### 6.1.6 Untoward reactions

Undesired complications after immunization, including abscesses or death, are tracked, since such incidents may discourage people from getting their children immunized, or indicate mistakes by vaccinators.

### 6.2 Divergent philosophies: DHIS 2 vs. RIMS

The two competing systems, DHIS 2 and RIMS, had quite different beginnings, as outlined in sections 4.3.1 and 4.2. Whereas RIMS was a specific solution to a limited problem, implemented by a contractor hired by a global organization, DHIS was a highly generic response to a situation of great social change, implemented by enthusiasts working with the grassroots. As the data standards will show, these aren’t just abstract categories – the social and political background of the systems inform just about every aspect of their designs.

In the discussion that follows, refer to the comparison of DHIS 2’s database design and that of RIMS, given in table 6.1.

#### 6.2.1 DHIS 2: Entity-Attribute-Value

The DHIS 2 database follows a design pattern called Entity-Attribute-Value (EAV). The goal of this design is to allow the user to define his or her own data elements. By adding entries to the data-element table (table 6.1a on the next page), the user may add new vaccines, new demographic groups or even completely different kinds of information. The data-value table (table 6.1b on the facing page) simply refers to each data-element identifier; the software looks up the data element proper on the fly without the user ever seeing the identifier.

This allows profound flexibility and essentially makes DHIS 2 fundamentally a statistics-collection tool tailored for health information. No programmer has to be involved in order to change what data is collected. For more on this philosophy, see section 4.4.

---

1For readability, foreign keys have been replaced by names in in this text, and names have been shortened. Values are invented.
### 6.2. DIVERGENT PHILOSOPHIES: DHIS 2 VS. RIMS

(a) A simplified representation of DHIS 2’s data-element table.

<table>
<thead>
<tr>
<th>Data element id</th>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>487</td>
<td>OPV 0 &lt; 1 year M</td>
<td>Number</td>
</tr>
<tr>
<td>490</td>
<td>OPV 0 &lt; 1 year F</td>
<td>Number</td>
</tr>
<tr>
<td>43</td>
<td>TT1 Pregnant women</td>
<td>Number</td>
</tr>
<tr>
<td>33</td>
<td>Spontaneous abortions</td>
<td>Number</td>
</tr>
<tr>
<td>137</td>
<td>HIV+ male</td>
<td>Number</td>
</tr>
<tr>
<td>157</td>
<td>Facility has neonatal care unit</td>
<td>Yes/No</td>
</tr>
<tr>
<td>1151</td>
<td>Type of LLTrn training</td>
<td>Text</td>
</tr>
</tbody>
</table>

(b) A simplified representation of DHIS 2’s data-value table.

<table>
<thead>
<tr>
<th>Data element</th>
<th>Source</th>
<th>Period</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>487</td>
<td>Anand PHC</td>
<td>Jan 08</td>
<td>43</td>
</tr>
<tr>
<td>490</td>
<td>Anand PHC</td>
<td>Jan 08</td>
<td>51</td>
</tr>
<tr>
<td>43</td>
<td>Anand PHC</td>
<td>Jan 08</td>
<td>25</td>
</tr>
</tbody>
</table>

(c) A simplified representation of RIMS’s immunization-session table. “<1” refers to infants below one year of age, with male, female and total numbers. “Preg. w.” refers to pregnant women.

<table>
<thead>
<tr>
<th>State</th>
<th>District</th>
<th>PHC</th>
<th>Mon.</th>
<th>Yr</th>
<th>Vacc.</th>
<th>Antig.</th>
<th>&lt;1 M</th>
<th>&lt;1 F</th>
<th>&lt;1 T</th>
<th>Preg. w.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guj.</td>
<td>Anand</td>
<td>AND1</td>
<td>Jan 08</td>
<td>OPV</td>
<td>0</td>
<td>43</td>
<td>51</td>
<td>94</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Guj.</td>
<td>Anand</td>
<td>AND2</td>
<td>Jan 08</td>
<td>TT</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>25</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.1: A comparison of DHIS 2 and RIMS database design.
The downside to this flexibility is reduced speed (for users) and increased complexity (for developers). The database management system (DBMS) must look up information from more tables than in a more traditional snowflake/star pattern (introduced next). For everyday operations this may not be noticeable, but operations working with large amounts of data take more time.

For developers, the price of the increased flexibility is that programming against the database must be done in a much more abstract fashion. The programmer cannot think in concrete terms like “vaccines” and “PHCs”, rather, he or she must write queries that work with “data elements” and “organization units”. The resulting queries are often complex and prone to error, although they only must be written once. Furthermore, the generic nature of the table means that the DBMS cannot perform the same degree of error checking.

### 6.2.2 RIMS: Snowflake

RIMS loosely follows a more traditional approach to database design, with its relational schema. Here, a table only contains a specific type of information: there is one table for vaccines, one for PHCs, etc. A new vaccine type can be added to the vaccine table, but no other kind of information goes there. See table 6.1c.

The various tables are organized in a snowflake (or star) pattern. The pattern describes a central fact table supported by dimension tables [GMUW08, 467]. The snowflake in figure 6.3 illustrates how the tables are connected: the immunization-session table (the fact table forming the nucleus of the snowflake) is surrounded by dimension tables such as PHCs, vaccines, states and districts.

This pattern is obviously more limited in terms of what kind of information the user can add. This is not surprising, in light of RIMS being an artifact of a vertical program with data elements from a global standard defined by the WHO. There are advantages, however; for instance, the database management system (DBMS) can automatically check for many errors and help maintain database integrity. If a table of immunization sessions refers to a vaccine with the code vac04, the DBMS can check that vac04 actually exists, and even prevent its accidental deletion. This feature is called foreign-key constraints. Databases are usually also faster and smaller.

Although there is nothing intrinsically wrong with this approach to database design, the RIMS database did not utilize some of its advantages. For instance, the designers skipped foreign-key constraints and other in-
6.3 RIMS schema issues

Throughout analysis of and interaction with the RIMS database, I came across a number of peculiar legacy features.

6.3.1 Denormalized

The most prominent is the startling lack of normalization. To briefly explain the concept, a database schema is denormalized if the following is true:

If two rows agree on attributes \( A_1 A_2 \ldots A_n \), then they also agree on another list of attributes \( B_1 B_2 \ldots B_n \). \[\text{GMUW08, 68}\]

In database theory, this is known as a functional dependency and is denoted \( A \rightarrow B \). In other words, A and B are redundant. For instance, an attribute country depends on city, since a specific city will be associated with the same country in every row. The attribute country is thus redundant and the association with the city should rather be stored in a single row in a separate table. Examples from RIMS will follow shortly.
This redundancy meant that I had to take greater care when loading data into the system, since I had to insert correct and consistent data redundantly in several locations. I found the following denormalized information in my analysis:

**Gender totals:** In addition to numbers for males and females, a total of the two genders was stored in the database. The functional dependency is $MF \rightarrow T$.\(^2\) The user interface allowed input of this data for children over 1 year (but not younger children). This could be trivially calculated by the software, and storing it increases the risk of error, especially when calculation is done by humans (which it often was).

**Vaccine balances:** The database not only stores the opening balance of vaccines and how many were consumed, but also the closing balance. The functional dependency is $OC \rightarrow B$.\(^3\) This could also be trivially calculated.

**District and state codes:** The database contains a table for each PHC, assigning it to a specific district and state. But there also is a district table, also assigning the district to a state — in other words, a district is assigned to a state in two separate tables, increasing the risk of error. The functional dependency is $P \rightarrow S, P \rightarrow D$.\(^4\) If that were not enough, the same PHC-district-state duplication is repeated in most other database tables. Now, district and state borders may not change very frequently, but erroneous information can be corrected only with some difficulty.

**District and PHC Codes:** Interestingly, the district code is merged into the PHC code, so that, for example, Anand has code “AND” and a PHC in Anand might have code “AND12”. This breaks the “first normal form”, which requires that each attribute is atomic: that it represents one thing and one thing only [GMUW08, 103].

As mentioned above, the software allows totals to be entered by operators — meaning that the denormalized state of the schema was not merely an academic consideration, but a present danger. The user interface was inconsistent with regard to which data elements accepted totals and which allowed only gendered data.

---

\(^2\)Where $M$ is male, $F$ is female and $T$ is the total.

\(^3\)Where $O$ is opening balance, $C$ is consumption and $B$ is the closing balance.

\(^4\)Where $P$ is the PHC, $S$ is the state and $D$ is the district.
6.3. RIMS SCHEMA ISSUES

We can speculate that performance, flexibility or external constraints were the reasons SOL chose a denormalized schema — this is a common rationale for denormalized databases. A hypothetical speedup might be achieved by not having the DBMS look up information in several tables when it can be collected into a single table, for instance, when tallying immunizations for the whole country. However, the referenced tables of states and districts are rather small — there are, after all, only 35 states and territories in India, and a few hundred districts — so I believe the advantage would not be great.

External constraints could have forced a denormalized design. For instance, PHC codes may have been determined long ago and not be unique, so that the code could only be used in conjunction with codes for districts and states. This was not the case, however - the vendor determined all codes, and they are all unique.

Finally, the designer may have intended a flexibility to choose between gendered and non-gendered data. In other words, some data may have been available for males and females, while other data only has the total number available. This would be a valid reason for denormalization, but the uneven implementation in the user interface suggests that this feature was accidental.

6.3.2 Unmarked foreign keys

Although the database tables contain plenty of references to other tables (such as vac4 referring to a specific vaccine), these were not marked as “foreign keys”, the database term for such references. This means that the DBMS (such as Microsoft Access or Microsoft SQL Server) cannot check that these references are valid. For instance, the vaccine vac4 may be inadvertently deleted, rendering data that depends on it useless.

Furthermore, these references were not numbers automatically generated by the DBMS, but rather text strings (such as “vac4”) defined by the vendor at design time. Although they are slightly more readable, such strings take up more space and are slower in use.

6.3.3 Data types

During the troubleshooting phase described in section 7.3.2, I discovered that certain table columns had unexpected data types. A “data type” refers to the general type of data, such as “text string” or “number”, but also some specific details, such as “byte” (a small number, less than 255) or “floating
point” (a large number). Presumably to save space, RIMS designers stored month numbers (which would always be 12 or less) in a single byte, rather than using a regular size number. Considering the vast space wasted by using textual references (section 6.3.2 on the previous page) and redundant district and state codes (section 6.3.1), this choice seems puzzling.

However, strictly speaking it was not an incorrect decision, and the designers could not have predicted that this choice would cause problems for Jackcess during this project.

### 6.4 Reduction of Form 6

In section 2.5.2, the Indian situation was compared unfavorably to post-Apartheid South Africa. Where South Africa had created minimal, flexible standards that scaled to the whole country (section 4.3.1), India relied on a single standard with 1500 data elements, with the same data elements for cold, mountainous regions such as Himalaya and tropics like Kerala.

During the winter of 2008, HISP project leaders worked closely with the Ministry of Health in India to reform the set of forms used to gather information in the public health sector. Inspired by the flexible standards philosophy (section 2.5), they reduced Form 6 from 1500 data elements to 50 for Subcentres and 100 for Primary Health Centres. Some population statistics were transferred to yearly surveys, and the male/female distinction was removed when redundant with other forms.

The new forms were piloted from mid-February 2008 in several states.

### 6.5 Summary

The data for the Universal Immunization Programme is mostly statistical, with the exception of cold-chain equipment status, which is inventory. The information bubbles up through several levels, with Primary Health Centres reporting on paper to districts, who fill databases and report to the states and ultimately the national government. The standard forms are vast and somewhat redundant, but in 2008 the government initiated a major effort to reform them according to flexible standards theory.

Although both DHIS 2 and RIMS can be used for UIP data, the two health information systems are founded on strikingly different philosophies. DHIS 2’s EAV approach allows the user complete flexibility in defining the data to collect, at the expense of speed and ease of comprehension.
It is also focused on statistical information. RIMS’s more traditional relational style is easier to understand and potentially more robust, but less flexible. However, the RIMS schema had many shortcomings so that the advantages of the relational style were not realized. This caused me some trouble when building the gateway, as described in the following chapters.
Tasty Gujarati lunch, delivered every day to the office.
Chapter 7

Pentaho gateway

As explained in section 5.1, the HISP leadership desired to gain experience with integration software, especially of the Extract-Transform-Load variety. After careful analysis of the existing data standards, my team therefore proceeded to implement a Pentaho Data Integration transformation that would read information from RIMS and transform it for use in DHIS 2. Unfortunately, communicating with the RIMS database ultimately proved futile. This chapter describes the process of trial and error in the work with the RIMS database.

Choosing immunization session information as the starting point, I began designing a transformation to extract data from DHIS 2’s MySQL database and load it into RIMS’s Access database. If the data could be transferred from DHIS 2 to RIMS and back again essentially unchanged, we would have proof that the process is safe.

The goal of gateway design was to have a button in the DHIS 2 interface that would initiate a Pentaho transformation. This transformation would connect to a RIMS database and extract data without further user involvement. Such integration is possible and relatively easy, since Pentaho and DHIS 2 both run on Java.

7.1 Creating a transformation

The following describes the transformation shown in figure 7.1 on the following page. See section 5.4.3 for an overview of transformation terminology.
Figure 7.1: The transformation intended to extract information on immunization sessions from DHIS 2 and load it into RIMS, as displayed in Spoon.
7.1. CREATING A TRANSFORMATION

7.1.1 Extract

The “Read immunizations” step sends a query to the DHIS 2 database to extract information on immunization sessions. The extracted information is implicitly sent to the next step. Table 7.1a on the next page shows the data as it looks right after reading.1

7.1.2 Prepare data

The next several steps prepare the data for merging with RIMS data. The modified data are shown in table 7.1b.

Select picks out the interesting columns from the data and does minor alterations. In particular, it duplicates the data element id, since DHIS 2 uses different IDs for vaccines and antigens, while RIMS associates them.

Flatten converts the data from the DHIS 2 rule, one data element per row, to the RIMS rule, with several data elements per row. (See table 6.1 on page 79 for another illustration of this.) It groups the data elements belonging to a certain health center and a certain month, and places them in columns, one after the other.

Add missing column adds a data element that did not exist in the DHIS 2 database, Immunization of infants over 1 year of age, in order to match RIMS’s schema.

Adjust fields selects, alters or removes fields (that is, columns). In this case, it simply removes DHIS 2’s periodid column.

7.1.3 Look up additional data

After the data has been prepared, we can look up information in the RIMS database to make sure the data are valid and the necessary PHCs exist there.

Convert PHCs replaces the codes used for PHCs in DHIS 2 (such as 1, 13, 49) with the codes used in RIMS (such as DNG1 or AND8).

PHC lookup looks up the health centers in the RIMS database and replaces the DHIS 2 codes with them.

---

1All values are fictional.
(a) The schema after reading from the database. Note that totals are not represented in the DHIS 2 database.

(b) The schema after preparing the data. Some data fields are omitted for brevity.

(c) The schema after transforming the data. Many auditing fields and a few data fields are omitted for brevity. The code “DNG1” refers to the first PHC in Dang district.

Table 7.1: A simplified representation of sample data for the transformation in figure 7.1.
7.2. **DIFFERING “DENORMALIZATION” TERMINOLOGY**

**Date adjustments** converts the date representation to the format used by RIMS. In DHIS 2, a period can have any start and end date, such as 1 January 2008 to 31 January 2008. RIMS, however, only accepts month-long periods and stores the month number and year. Therefore the next several steps have to pull out those numbers and filter out periods longer or shorter than a month.

**Vaccine and antigen matching** looks up the vaccines and antigens in the RIMS database and substitutes their codes.

### 7.1.4 Clean up data

The last few steps ensure that the database schema matches RIMS exactly, removing superfluous fields, reordering the remainder and adjusting their types (for instance, marking numbers and text as such). In addition, the total number of immunizations must be calculated. The transformed data are illustrated in table 7.1c.

### 7.1.5 Load

Finally, we may insert the transformed data into RIMS’s Microsoft Access database.

### 7.2 Differing “denormalization” terminology

I had some initial difficulty identifying step types that would convert from DHIS 2’s highly flexible data-element-centric database to RIMS’s fixed data elements. (See table 6.1 on page 79 for a reminder of the differences between DHIS and RIMS in this regard.)

What I needed was a step that would

1. read each line extracted from DHIS 2 and determine which data element it represented, at which health center it was collected and in which month

2. insert the data value into the correct column for the data element

3. group these data values by health center and month

It turned out that the “Row denormaliser” (called “Flatten data” in figure 7.1) step provided these features, although the name would not indicate
this. Denormalization would usually imply introducing redundancies, but no redundant data was introduced. What this step really does is translate from the Entity-Attribute-Value schema used by DHIS 2 (section 6.2.1) to the relational schema used by RIMS (section 6.2.2). This PDI definition of normalization was also implicitly used in conversation with other HISP members, even though they did not work with PDI, indicating that it is a widespread understanding of the word.

Once the correct step was identified, the rest of the transformation practically wrote itself. One thorny problem, however, remained: loading the transformed data into Access.

### 7.3 Loading data into Access

#### 7.3.1 ODBC

I attempted several approaches to loading the data into RIMS’s Microsoft Access database. First, I tried what would be the default approach, namely ODBC, a standardized interface for connecting to databases. PDI would connect to the ODBC service built into Windows, and this service would forward the connection to an instance of Microsoft Access running on the computer. Access would then interpret the queries I gave it and reply with data through the ODBC service to PDI.

Sadly, this method gave no results whatsoever, as the database connection always returned “General Error”: no specifics, nothing to work with. Therefore I had to abandon it and move on.

#### 7.3.2 Jackcess

In my next attempt I used Pentaho’s Access Output step, opening the Microsoft Access file directly. This also consistently returned mysterious error messages. The messages seemed to have little to do with the data I just had generated, often referring to incorrect data types and out-of-range numbers. However, they were specific enough to give me something to work with, and this time I had access to source code. I started digging deeper.

Upon examining PDI’s source code, I found that it uses the Java library called Jackcess (section 5.4.4) to interface with Microsoft Access databases. At first, I suspected that the mysterious error messages were caused by PDI using the library incorrectly, and consulted Jackcess documentation.

---

2See also a discussion of this terminology at the PDI wiki at [Bil06].
However, it soon became apparent that Jackcess at the time was in alpha state — a term indicating that the software is experimental and not yet recommended for general use.

It is not difficult to understand why Jackcess was so immature. The Microsoft Access data format was and is secret, and Jackcess did not yet read and write correct data consistently. In our case, the columns always came out in the wrong order, causing the data-type errors (when, for instance, Jackcess would try to force a text string into a number column). Obviously, data would be corrupt (in the wrong order) even if no data-type errors had been thrown.

I reported this bug to the Jackcess bug tracker [Sto08]. The lead developer, James Ahlborn, replied with good news and bad news: the next version of the software would not have this specific bug, but support for large databases was still in the pipeline. I considered investigating this and contributing such support myself, but in a rather urgent situation with a waiting customer, I did not see that as a realistic option. Interestingly, I would not have had to wait long: Support for large databases was added a few weeks later, but only after the Pentaho integration project had been abandoned [Dav06].

### 7.4 Data transfer using the RIMS user interface

Time was growing short and our customers in Gujarat were waiting. We needed to look to routes other than exporting directly into an Access database. Instead, we decided to try importing through the RIMS graphical user interface, on the assumption that the built-in features would be able to handle well-formed data without trouble.

The expected downside was that this would require more steps and manual intervention, and also negotiating user-interface bugs. Users would have to click through several layers of menus and dialog boxes, and several of these screens had defects that made them difficult to work with.

In order to test the import/export, I exported RIMS data input by Gujarati health workers in early 2008 to the so-called “text format”, which is to say comma-separated values (illustrated in figure 7.2 on the following page). Then I imported the same data, unchanged, back into RIMS using the built-in import functionality.

Surprisingly, using the built-in import/export functionality turned out to be just as difficult as direct database export. When I attempted to export data from RIMS in “text format” (comma-separated values) and then
import the same data back into RIMS, the software threw mysterious error messages, such as “Syntax error in FROM statement part” and “ErrorInfo.GetDescription failed with E_FAIL(0x80004005)”. Similar problems cropped up when exporting and import using XML, a common data interchange format.

Error messages that mention “syntax errors” (that is, using the programming language incorrectly) indicate defects in the RIMS software. That this occurred on freshly-exported data indicates that SOL had not tested this functionality thoroughly. With access to the source code, repairing the defects would most likely be trivial, but without it I simply had to look to other methods.

Eventually, I did find one method that did not crash the software: Import and export using the Excel and Access file formats. (One may speculate that these were the only file formats used by the RIMS developers.)

By this time, however, it had become clear to the project leaders that a solution which requires that degree of user intervention was unacceptable. It was at this point that postponing the Pentaho solution was first suggested. Instead it was proposed to prioritize the ad-hoc solution using a custom DHIS 2 web module that accessed the RIMS database directly. In this way, it was hoped that we could gain more control of the process and make it easier for users.

7.5 Source code and software access

As mentioned earlier, many of our problems could have been solved with access to the source code. With such access, I could have tried to fix errors in RIMS that prevent import and export. My team asked for the RIMS
7.6. SUMMARY

Source code from the WHO, who forwarded these requests to SOL, who had written the code in question. Unfortunately, despite repeated requests the contractor never provided the code, even though they were rehired to do so.

I also wished to test integration with the online version of RIMS (which is based on Microsoft SQL Server rather than Microsoft Access) to identify any problems at the earliest possible stage. Unfortunately, this software was also unavailable.

7.6 Summary

In the effort to employ Extract-Transform-Load software to integrate our two health information systems, it turned out that the Load phase was the weakest link. I tried inserting data into a Microsoft Access database by three methods: a standard ODBC connection, direct file manipulation through Jackcess, and manual file import.

None of them were successful: the ODBC connection consistently generated vague error messages, direct file manipulation was not yet mature and reliable, and manual file import was too work-intensive for daily use. Troubleshooting was difficult since I was not given access to RIMS source code, and the Microsoft Access database-file format is secret. Instead, an ad-hoc solution was attempted and is described in the next chapter.
A street merchant spending the warm spring night near his cart.
Chapter 8

Ad hoc gateway

The ad-hoc gateway was the last resort attempted to integrate RIMS and DHIS 2 after the failure of the ETL approach. This chapter gives a summary of the development process and challenges that we encountered.

8.1 Development process

After the Pentaho Data Integration solution had been abandoned, Bharath Kumar, a lead developer for HISP in India, wrote a DHIS 2 module that provided import and export through ODBC. The ODBC approach was tried in Pentaho, but threw mysterious error messages (section 7.3.1). Kumar’s module did not suffer from this problem; we did not find out why.

An alpha version of the module was written in a few days, supporting three tables: vaccine supplies, immunization sessions and vaccination. The rest of the work was completed at the customer location in Gujarat, where support for the rest of the tables (except cold-chain equipment) was added, and data elements missing from the DHIS 2 database were added. I added sophisticated error handling, automated creation of DHIS 2 data sets and optimized for speed. As with the Pentaho gateway, unnormalized data in RIMS made the work more difficult and time-consuming.

Representatives for the customer expressed a wish for phased interoperability of the new software, by providing links to online RIMS reports rather than duplicating them in DHIS 2.
8.2 Testing

Working with the staff of the Gujarat Ministry of Health, we laid a plan for testing export functionality. To test correctness, we decided to export data from RIMS to DHIS 2, then import it again and compare reports from before and after. The data was from April 2007 for three districts (Valsad, Anand, Mehsana) and their PHCs. Once the test data worked, live, current data would be tested.

I also carried out a usability test of various input methods in DHIS 2 — with data elements were grouped or ungrouped, or with a data-entry form that resembled the paper form.

8.3 Cold-chain equipment

Although the information on cold-chain equipment was analyzed earlier (section 6.1.5), it was only later that I realized that this information had no natural home in the statistically-oriented DHIS 2. The equipment information was inventory-oriented and fundamentally different from statistical information on immunization. Furthermore, communicating this problem to users was difficult, since they consider the distinction between statistical data and inventory data to be irrelevant. “It’s all information!” was an assertion I often heard!

When I left Gujarat, cold-chain equipment remained the main challenge. Several approaches were considered, including a new DHIS 2 module for inventory information. However, the solution implemented was a “line listing” feature that provided a simpler implementation, and was used for other features as well. With this addition, DHIS 2 could replace RIMS and import its data.

8.4 Other data issues

Many data elements in the UIP form were also found in Form 9 (the redundancy the project aimed to remove). At first, examining the DHIS 2 database seemed to indicate that Form 9 was missing a number of data elements found in RIMS. Later, however, a paper printout to which I was given access showed that most elements actually were in place, but age ranges present in RIMS were missing in DHIS 2, leading to confusion.

I assumed we could get away without adding the redundant “total” fields for male and female (mentioned in section 6.3.1), but it turned out
that several PHCs of the Mehsana district (Jhotana, Sudasana and Bamosana) used totals, not genders, exploiting what I thought was “accidental feature” on the part of the RIMS designers (section 6.3.1 on page 83). Therefore I had to add the redundant data elements to DHIS 2, and automatic totaling became more complex, even though there are DHIS 2 features to help calculation.

8.5 Summary

Initial development on the gateway proved easier than the ETL solution. However, the fundamental difference between inventory information and statistical data meant that additional work was required on DHIS 2. This distinction proved difficult to communicate to users. Some anticipated RIMS problems surfaced. Despite this, the solution was implemented and eventually used to replace RIMS with DHIS 2.
Part III

Discussion and Conclusion
In this part, I will analyze the existing systems encountered in this project and the results of my work, as well as some thoughts on open source and open standards in a setting transitioning from legacy systems.

Chapter 9 compares this project with theory on organizational integrations.

Chapter 10 analyzes the situation before the project started, looking at DHIS and RIMS as parts of information infrastructures. It looks at RIMS from a different angle, as a legacy system, and evaluates our proposed gateways according to II theory.
Kovalam beach.
Chapter 9
Organizational Integration

This chapter revisits the frameworks laid out in [WW04] and [Car04], analyzing the project according to those theories. I will also compare the two frameworks.

9.1 The organizational domain

[WW04] recommends upfront analysis and proactive handling of organizational issues. We were prepared for some issues, but we also had some surprises. The framework also has value when applied in retrospect, which I will do here.

In figure 9.1 on the following page I have illustrated how the various domains interacted with the project:

1. The Government of India (GOI) is used as a starting point because its goal of universal immunization of the population was the impetus for the work that followed.

2. The Universal Immunization Programme (UIP) in collaboration with the World Health Organization was a means of working toward the goal, but unfortunately led to fragmentation of Indian health information systems.

3. HISP decided on a strategy of integration to address this issue, and explored Pentaho as a platform for integration.

4. Sadly, technical issues such as lack of source code and database problems were not addressed by the WHO contractor in Geneva, perhaps as a result of politics, or even strategy, on their side.
Figure 9.1: An illustration of this project’s meander through the various domains. It started in the power structures of the GOI (circled), or Government of India. We tried integration between the two systems, but ended with scrapping the legacy system and using DHIS 2.
5. As a result, integration efforts were abandoned and a freshly-installed DHIS 2 strategy (“cut and run” or “revolution” in section 2.7) was used instead.

Let us take a closer look at the organizational and strategic domains of the project:

9.1.1 Structure and culture

In [WW04], seven aspects of culture are identified. Here, I will emphasize two, with respect to the organizational issues introduced in section 5.6. Since the project did not reach the implementation phase, I did not put the advice we received into practice.

Innovation and risk taking

The careful, non-disruptive strategy to structural change encouraged by the UIP in section 5.6 is pragmatic when dealing with organizations that do not usually react quickly to changes in the environment.

Top management contact and reward orientation

The peculiar situations described by the UIP representatives had middle managers reporting one statistic to politicians while actually working with another, more well-founded figure. This can be seen as fear (of reprimands, or losing one’s job) or reward orientation, with rewards given for higher indicators, rather than correct indicators.

9.1.2 Power and politics

HISP is no stranger to politics and power. As mentioned in [BH02, 119] and section 4.3.1, “HISP has never been fully accepted by official structures, but never fully dismissed either.” In this project, too, we encountered this ambivalence in several respects.

One issue that was mentioned early on by UIP people (section 5.6) was the necessity of identifying, retaining and making the most of key personnel in the field prior to implementation. This is obviously important because of their skill and experience, but it is also a way to approach already-existing power structures and getting the people in charge on our side.
In the end, the power that most dramatically impacted this project was that wielded by the contractor that developed RIMS, even though this contractor was the most physically distant (Geneva). As it became clear that access to RIMS source code would be beneficial to the project (sections 7.4 and 7.5), my frustration at the contractor’s lack of response grew.

It may seem quite surprising that help was not forthcoming, since the customer, WHO, was obviously interested in better health services in developing countries. However, reading how [Mar83, 441] recommends “self examination of interests, motives, payoffs, and power bases” (quoted in [WW04]) and looking at it from the contractor’s point of view, we might begin to understand. The contractor had no incentive to revisit a project he had finished and do work he probably would not be paid for, especially considering HISP provided a solution, DHIS 2, that competed with his. Why help a competitor?

If we could redo the project, it may have been beneficial to have responded to the political issue using powerful allies on our side. Escalating the problem to higher-ranked people in the organization might have been more efficient than working around it.

The power advantage gained by the Geneva team forms a striking contrast to the complete lack of power games by the Jackcess developers. The developers shared their work freely and responded to requests for help. The lack of competition in this open-source situation afforded free flow of information. Figure 9.2 illustrates how open-source software touches the domains of strategy (section 9.1.3) and power.

9.1.3 Strategy

Many of the problems we encountered during the development process — and most of the serious ones — could be traced to proprietary software and standards in some way. Section 9.1.2 dealt with the power relations of the situation, but it also has a strategic side (section 2.1).

The legacy system, RIMS, frequently gave errors indicating missing data validity checks and programming mistakes (section 7.4). These would be simple to fix if source code had been open.

The database platform, Microsoft Access, had a closed database file format which made reading from and writing to the database an experience in trial and failure (section 7.3.2).

Compare this to the experience using Jackcess, whose defects were fixed by developers with similar problems outside the project.
Figure 9.2: Open-source software affected the technical, strategic and power domains in this project.
In sum, this suggests that integration through gateways can be simpler when a legacy system has source code available and uses open standards. One could argue that investing in such systems would be a more “future-proof” investment than proprietary software, greatly reducing the risk of lock-in.

HISP’s encounters with Microsoft Access illustrate this from two different angles. First, it was decided almost a decade ago to move from the proprietary, Windows-only format to reduce lock-in, increase transparency, work more easily with the open-source software development world and be better positioned for developing countries (section 5.4.1). Second, Microsoft on their side has pursued the opposing strategy: closed software is supposed to build or strengthen their market share. Microsoft’s approach to developing countries involves lower-cost licenses rather than open standards.

9.2 Boundaries and novelty

The framework found in [Car04] is also useful in analyzing the case. Instead of a set of four domains, we have a stack of three layers (syntactic, semantic and pragmatic).

9.2.1 Boundaries

I experienced syntactic and semantic boundaries in several situations. A syntactic boundary was met when trying to communicate the difference between statistical reports and cold-chain inventory information to management (section 8.3). Management had no reason to know about or care about different types of data, distinctions that seem so important to engineers. As the [Car04] framework predicted, once the common lexicon (at the syntactic boundary) broke down, it was difficult to reconcile at the semantic and pragmatic levels. The other party did not understand the importance of this difference (semantics) and was not interested in waiting for development to finish (pragmatics). Notably, the syntactic boundary between humans was based on a syntactic boundary in the machine, since the DHIS 2 database lacked the “language” to represent inventory information.

The framework’s recommendation for a syntactic breakdown like this is to work on the common lexicon: sharing the categories of information (inventory and statistics). Then common meaning can be built, by explaining
the consequences of this difference. Finally, the shared interest in representing the inventory information despite the added cost would hopefully be identified, although this may be wishful thinking from a software engineer!

In another incident, the definition of “normalization” diverged between database theory, which I was familiar with, and use of the term by PDI and HISP programmers (section 7.2). Naturally, this represents a semantic boundary between the two groups. The framework recommends creating common meaning. We might go about this by agreeing on shared terminology (syntax) with other groups, or mentally translate between their definitions and ours. Shared meaning (semantics) can be achieved once the syntactic foundations are clear.

The biggest impact was at pragmatic boundaries. The foremost example is the diverging interests of RIMS developers and us which prevented sharing of source code. I attempted working around this at the syntactic level (programming integration solutions without source-code access). As predicted by [Car04], such tackling of issues at the wrong level was fraught with great difficulty and cost.

9.3 Evaluation

The frameworks appears to be useful in preparing for organizational challenges. The political domain from [WW04] predicted some of our gravest problems, and [Car04] presented solution mismatches that could have helped us solve problems. However, to prepare adequately, more exhaustive analysis than the frameworks alone (perhaps using some of their suggested approaches, such as that of [Mar83]) would have been necessary to identify potential weak points. It then becomes a question of how much should be invested in upfront analysis rather than starting work and dealing with contingencies as they occur.

9.4 Synthesis

The two frameworks are quite different, in that one focuses on a single domain among several equally valuable domains, and the other brings together several organization theories and strategies in a stack. However, using them together gives us a wider vocabulary to employ when addressing organizational issues.

[WW04] claims that organizational issues have had much less attention than technical aspects, while [Car04] says that pragmatic boundaries
have more novelty (more “unknowns”) than syntactic and semantic boundaries. I propose that the neglect of the organizational domain is due to the added novelty (or uncertainty) introduced by human organizational politics and strategy makes for less common knowledge and less understandable boundaries than the purely syntactic and semantic ones found in the technical domain. The application of boundaries to domains is illustrated in figure 9.3.

The frameworks are certainly compatible when it comes to the advice to treat problems belonging to a domain or boundary within it. Whether viewed as the political domain or a pragmatic boundary, dealing with the power relationships between the Geneva contractor and HISP would have been better handled up there than in the technical domain or the syntactic/semantic boundaries.

In this day and age, the technical process of integration is quite well understood; a small team might finish programming an integration solution quickly (touching mostly on syntactic and semantic boundaries). Systems and strategic thinking (such as HISP’s focus on open source, open standards and generic solutions) are more unpredictable: they involve both heterogeneous complex adaptive systems and time frames spanning months or years. Meanwhile, social and political issues (with pragmatic boundaries) are quite difficult to predict and model and can unexpectedly bring an integration process down, as in our case.
9.5 Summary

The analytic frameworks proved useful for the organizational issues of this project. In particular, the greatest challenges we faced followed the axes of “power and politics” and “strategy” in [WW04] and the “pragmatic boundary” in [Car04]. This fits the predictions of each framework, suggesting their usefulness in upfront analysis of future organizational issues. Indeed, the frameworks work well together, and I found it meaningful to apply the concept of syntactic, semantic and pragmatic boundaries from [Car04] to the domain thinking presented in [WW04].
Autorickshaws on the streets of Kovalam.
Chapter 10

DHIS and RIMS as infrastructures

This chapter attempts to explain how DHIS and RIMS interplay with their respective infrastructures, using information infrastructure (II) theory as laid out in section 2.4. It lays out the shortcomings of the current situation according to information infrastructure theory and evaluates the legacy-change process.

10.1 DHIS as an II

HISP has enjoyed close ties to the research community around information infrastructures, and several project activists contribute to the field. Naturally, II concepts, such as gateways, have been applied to DHIS and the other HISs the project works with. (More on gateways in section 10.3.3.)

It should be obvious by now that DHIS has an *installed base*: we find established installations as well as prototypes and development in many countries on several continents. It is *shared* in the sense that data entered in one installation can be shared with any other installation of DHIS (and even other systems, as we shall see shortly). Interoperability and backward compatibility occupy a central place in DHIS’s history, most markedly with the transitions from version 1.3 to 1.4 and 1.4 to 2.0 — the latter involving a transition from a Microsoft Access platform to a generic SQL platform.

Moving data between DHIS 1.4’s Access database and a DHIS 2.0 installation running, for instance, MySQL is simple when using DHIS 2.0’s built-in import/export module. This module is an example of a gateway that greatly eases the transition from an older version of a system to a newer
version, much like how this project eases the transition from RIMS to DHIS.

Integration with other health information systems has also been a priority, with a multitude of import/export options built into the software, including such options as IBM DB2 as well as the more common XML and CSV. Some work has also been done toward aggregation of data from OpenMRS and other electronic patient journal systems. As such DHIS can be considered both a infrastructure in its own right and a part of a larger health information infrastructure.

The integration with other systems and between various versions of DHIS already makes it a heterogeneous system — a part of an ecosystem of infrastructures (sometimes called “ecology of infrastructures”). Additionally, the infrastructure itself is varied: it has diverse activities (data collection, reporting, analysis), different kinds of information (text, numbers), various standards (the database schemata, export/import formats) and people (information collectors, developers, instructors and support staff).

DHIS evolves constantly, with new features being developed by teams around the world. It is open: new installations are continually added to the infrastructure; some disappear due to reasons such as unsuccessful prototypes, lost bids or transitions to other prototypes.\(^1\) An “open infrastructure” does not necessarily entail “open source”. However, freely available source code does mean that unexpected stakeholders and competing vendors may be added to the infrastructure outside the control of HISP. The infrastructure could thus grow without HISP’s acting as a gatekeeper, at least in theory.\(^2\)

### 10.1.1 Flexible standards

As mentioned in section 4.3.1, DHIS has been built around a hierarchy of standards from the outset. There is a minimal data set that is used by the entire country (or in the case of WHO programs, the world), with increasingly larger sets used by lower levels. Many infrastructure members collect the “need to know” information, while fewer collect the “nice to know” information. This creates great flexibility because standards can be combined

---

\(^1\)Sadly, some of these situations were due to bribery by competing vendors. Corruption is as rampant in HIS contexts as elsewhere.

\(^2\)There are suspicions that certain competing vendors have provided a paid, relabeled version DHIS 2 with closed source code. Although understandably unpopular with activists, such closing is allowed by the BSD-style license that rules DHIS 2. These have not attempted to work with the official versions of DHIS 2, however, so they are not part of the infrastructure.
10.2. RIMS AS PART OF AN II

as necessary.

For example, one state in India may choose to use DHIS to collect information for the Universal Immunization Programme (UIP) together with information mandated by the Ministry of Health (MoH), while another collects UIP information elsewhere and collects MoH information plus additional data elements of their own choosing.

10.2 RIMS as part of an II

RIMS shares many of the characteristics of DHIS and works in much the same way. However, as the product of a vertical program (section 4.1), it illustrates the problem of fragmented information infrastructures: information does not easily flow between the UIP’s RIMS and other parts of the Indian health system.

The most significant problem in this case, however, has been that development was essentially abandoned: RIMS no longer evolves. It is still open for new members, however, using knowledge that existing users can give to newcomers. New stakeholders, developers and vendors, however, have no way of taking part.

If limited to its technical components, the infrastructure around RIMS was less shared and heterogeneous than DHIS 2 simply because export/import functionality was unreliable and partially defective — even between installations of RIMS itself. However, in one sense it was more heterogeneous than DHIS 2: it tracked more types of information. It included both statistical information on immunization and inventory of cold-chain equipment and vaccine supplies (see section 6.1, especially subsection 6.1.5).

10.2.1 Lock-in

In section 2.4.1, we learned about lock-in situations that may occur in information infrastructures. The cost of switching to DHIS 2 was increased by the closed source and information formats - the missing access to source code and the closed Microsoft Access database format.

10.2.2 Inflexible standard

As implied above, there is no room for variation or combination of standards in RIMS. It collects data following a single standard, that of the UIP. We may therefore call it an inflexible standard, making RIMS less adaptive as a system.
10.3 Legacy change through gateways

In this section, I will review the project in the light of legacy theory and information infrastructure theory, particularly regarding gateways. I will evaluate the success of our gateways measured against the original design goals.

10.3.1 RIMS as a legacy HIS

Our legacy systems were much newer than those encountered by [Sko03, BLWG99]. They ran on modern computers, and in some cases — such as the central installation of Microsoft SQL Server in each state — used up-to-date, industrial-strength platforms. However, installations in district offices used the much smaller-scale Microsoft Access solution, which had its own host of problems, as described in section 5.4.1.

Furthermore, RIMS itself suffered from a number of [BLWG99]’s legacy problems:

Lack of documentation: Although I had access to a user manual, the inner workings of RIMS were a mystery. Only through extensive analysis of database schemata and frequent consulting with health experts did I achieve a working understanding (section 6.1).

No source code: The “black box” that RIMS was could be maddening when I was confronted with bugs and crashes that conspired to almost completely prevent import and export. With access to the source code, I could have traced the faults and fixed them (section 7.5).

Lack of interfaces: The import/export functionality was defective, as mentioned (section 7.4). I had to use the undocumented database structure to construct appropriate queries for retrieving information.

General poor state of system: There were numerous inconsistencies in the RIMS database, as mentioned in section 6.3.

These findings suggest that [HMS99] is still relevant: as an industry, we still have some ways to go.

10.3.2 Legacy change

In our case, changing work routines of the staff was less challenging than in many legacy situations, because the essential functionality of DHIS and RIMS was the same.
Throughout the process, we followed what might be termed “parallel operation” in section 2.7, since import and export would be used until the system could be trusted. In this case, it is the gateway that is not yet trusted, not the new information system, as theory suggests.

In information infrastructure terms, we have followed the “fast evolution” strategy. A gateway links the old and the new infrastructures together, meaning that no “flag day” is necessary: each node in the infrastructure (be it headquarters of a district, a state, or even the entire Universal Immunisation Programme) may choose when to switch to DHIS 2.

10.3.3 Closing gaps

Gateways

A gateway can be the final solution, as with the Nordunet plug in [Han01] (section 2.5.1). That solution evolved into an approach that is still valid and in use today, two decades later. Our gateways, on the other hand, were not intended as permanent solutions, although hypothetically they certainly could be.

Rather, our gateways were intended as incremental measures (see section 2.7.1). The import/export solution was intended as a stopgap to test that data integrity was preserved, so that the switch could be made safely. Furthermore, the incremental approach allowed gradual implementation of benefits provided by DHIS 2, such as custom reports. The customer would benefit at an earlier date, and we could begin testing earlier. When everything worked fine, RIMS would be replaced by DHIS 2. (As it turned out, though, we had to fall back on the revolution strategy.)

Participation

The earliest development and implementation of DHIS 2 was highly participatory in nature (section 4.3.1), with the pilot group intimately involved in the process. Such participation was also recommended by [HMS99]. In contrast, health services in Gujarat were not involved in the RIMS development process (section 4.2). This need not mean that none of the state’s health services were involved, but that at the state level less ownership was felt for the project as a result. The duplicate data collection (one of [HMS99]’s gaps, mentioned in section 2.3.1) may have resulted from this lack of involvement.

The participatory approach was also followed in the current project. We
developers spent several weeks with the customer, interacting daily with a variety of stakeholders, including users, health experts and managers. It was easy for the developers to pay close attention to customer needs; unfortunately, it was also easy to get lost in the details of implementation and lose the big picture. The latter led to some degree of “scope creep”, where minor problems took up more time than warranted when I lost focus on the larger goals.

10.4 Gateway evaluation

Extract-Transform-Load software can be thought of as a generalized tool to develop gateways. By making gateway development more rapid, disparate infrastructures can be merged more easily. As this is something HISP does all the time, that type of solution is highly desirable. A generic ETL tool would make the two infrastructures far more change flexible (section 2.5) than an ad-hoc solution, since any change to standards could be accommodated with far less effort.

We wanted the following qualities in the gateway solution:

**Correct**: It is fundamental (even implicit) that data integrity is essential.

**Robust**: A reliable gateway would have to be thoroughly tested and support changes without risky source-code changes.

**Rapidly adaptable**: Reusing the solution in new contexts should be painless.

10.4.1 Pentaho

Although the Pentaho gateway was not completed due to certain immature components, it was clear from an early stage that rapid gateway development was feasible and realistic within the framework.

**Correct**: Data could be visually inspected at any point in the transformation, making the checking of integrity during development easy. The software is used by thousands of people, and thus receives much stress testing.

**Robust and flexible**: No source code had to be modified; everything could be programmed in transformations.
10.5. SUMMARY

Rapidly adaptable: The visual design tool, Spoon, was relatively simple to understand and use, and the gateway could quickly be modified. However, errors were not as transparent; often “Generic error” messages would pop up with no further explanation.

10.4.2 Ad-hoc solution

The apparent complexity of an ad-hoc solution was higher since the developer must control many more aspects of the solution. However, the solution as a whole was simpler, so troubleshooting of low-level database errors was actually easier. All in all, though, modification was quite difficult, and an ad-hoc solution does not receive nearly the amount of testing that an open-source framework does, so the solution was less robust overall.

Correct: Inspecting data could be done with relative ease by developers, and the import interface provided users the opportunity to check for errors. As an ad-hoc solution with few users, it would not receive as widespread and thorough testing as a popular framework would.

Robust and flexible: Any changes required modifying the source code and taking the risk that something would break and errors would be introduced.

Rapidly adaptable: The solution was not easy to apply to other integration projects because it was not generic in nature.

10.5 Summary

This chapter described DHIS 2 and RIMS in their information infrastructures context and applied relevant theoretical concepts. Although the legacy system, RIMS, ran on modern computers, it met several criteria of legacy health information systems: it lacks documentation, source code and interfaces, and is generally in a poor state. This supports [HMS99]’s decade-old observations.

We followed strategies that may be named “fast evolution” or “parallel operation” according to different theories, using gateways to gradually implement the new system. To reduce rationality-reality gaps, we let users participate throughout the project.

When comparing the two integration approaches, I found the generic ETL solution to be trustworthy, robust and flexible. However, due to problems integrating with Microsoft Access, it was less adaptable. The ad-hoc
solution was inferior in every way, except that it provided transparency that allowed us to adapt it to Microsoft Access. Therefore it won out in the end despite its problems.
10.5. SUMMARY
Palm trees in Kerala as seen from a house boat.
Chapter 11

Conclusion

In this chapter I will draw general conclusions from the action-research project.

**Research objective** Explore the impact of integration using open-source ETL software in a closed-source legacy setting.

Overall, Pentaho Data Integration showed promise in the Spring of 2008, as data transformation programs could be more rapidly and comfortably developed than the previous ad-hoc approach.

That the software was open-source proved a double-edged sword. On one hand, open-source software is much easier to troubleshoot, so that root causes of my troubles were eventually found. The opportunity to use components from the vast open-source Java community gave increased functionality.

On the other hand, use of such components presented the risk that they were immature and not ready for general use. However, I was positively surprised by the willingness and speed with which the community responded to and solved the technical issues at hand.

It became clear that open-source ETL is not a silver bullet for closed-source and legacy software. These types of software still can and will pose problems that are not necessarily easy to solve, as seen in the answer the first research question:

**RQ1** How can closed source and closed standards affect an organization’s integration process?

In contrast to the open source code of PDI, the closed source code of RIMS meant that serious errors could not be investigated. This limited our options and unnecessarily delayed the integration project.
As for closed standards, more options were limited by the secret Microsoft Access file format. This hindrance was the cause of the immaturity of the components mentioned above. The case study provides a compelling demonstration of the strategic weaknesses of committing to black-box programs and standards. And beyond that, that strategy opens the door for power games, such as the following:

**RQ2** How can power, politics and culture in an organization impact the integration process?

Even closed source code may be shared, but this was not the case here. When withholding information and source code is possible, organizations run the risk that the actors controlling it wield that power to their own ends. In this case, it delayed and complicated the integration process.

**RQ3** Can user participation in information-system reduce gaps between plan and reality?

Data collection was limited on this point, but there are signs that lack of participation was a factor in the failure of RIMS. In this project, issues such as incompatible cold-chain data were discovered as a result of user participation after having slipped through developer analysis.

**RQ4** Compare ETL software to ad-hoc solutions for integration.

I found the ad-hoc solution inferior in terms of being trustworthy, robust and flexible. However, the added transparency it provided made troubleshooting easier and ultimately more successful.

### 11.1 Contributions

This thesis has explored frameworks for analyzing organizational integration and applied them to a mixed environment of open and closed standards and source code. It has synthesized frameworks found in [WW04] and [Car04], suggesting how they might work well together.

On the practical level, HISP has gained experience with ETL tools and their application to legacy integration challenges. We have explored topics of power and politics in the context of HISP, and applied frameworks for organizational analysis that may ease future projects.
11.2 Further work

It would be interesting to test the predictive and problem-solving value of the organizational frameworks used here. It may be that they would help future integration projects to run more smoothly.

Integration with other open-source software would form an instructive contrast to this case. Would we run into the same problems, no problems, or something completely new?

Lastly, I would suggest further exploration of participation as a way to reduce rationality-reality gaps.
Appendix A

Abbreviations

ANM  auxiliary nurse midwife
BI  business intelligence
BSD  Berkeley Software Distribution
CSV  comma-separated values
DBMS  database management system
DHIS  District Health Information Software
EAV  entity-attribute-value
ETL  extract-transform-load
GNU  GNU’s Not Unix
GoI  Government of India
GUI  graphical user interface
HIS  health information system
HISP  Health Information Systems Programme
HMIS  health management information system
HTML  Hypertext Markup Language
IDE  integrated development environment
II information infrastructure (theory)
IS information system
MoH Ministry of Health
NGO non-governmental organization
ODBC Open Database Connectivity
PDI Pentaho Data Integration
PHC Primary Health Centre
RIMS Routine Immunisation Monitoring Software
RTF Rich Text Format
SC Subcentre
SOL (unknown; a WHO contractor)
SQL Structured Query Language
UIP Universal Immunisation Programme
WHO World Health Organization
XML Extensible Markup Language
Bibliography


[Bil06] Bill Karwin. Comment to blog post "Key-value madness". http://www.ibridge.be/?p=15#comment-90, 6 November 2006. 92


BIBLIOGRAPHY


[KM05] Stephen Kemmis and Robin McTaggart. Participatory Action Research: Communicative Action and the Public Sphere, chapter 23. In [DL05], 2005. 11, 45, 46, 47
BIBLIOGRAPHY


[Sta05] Robert E. Stake. *Qualitative Case Studies*, chapter 17. In [DL05], 2005. 48


