Proceedings of the APSEC'97/ICSC'97: Joint 1997 Asia Pacific Software Engineering Conference and International Computer Science Conference. Hong Kong, China. 2-5 December 1997. pp 312-320, Ed. IEEE Computer Society.

Legacy Systems Migration - A Method and its Tool-kit Framework

Bing Wu, Deirdre Lawless, Jesus Bisbal,
Jane Grimson, Vincent Wade
Computer Science Department,
Trinity College, Dublin, Ireland.
Email: {*name.surname*}@cs.tcd.ie

D O'Sullivan[1], Ray Richardson[2]
Broadcom Éireann Research,
Dublin, Ireland.
Email: [1] : dosullivan@ broadcom.ie;
[2] : rr@broadcom.ie

Abstract

The problems posed by mission-critical legacy systems - brittleness, inflexibility, isolation, non-extensibility, lack of openness etc. - are well known, but practical solutions have been slow to emerge. Most approaches are "ad hoc" and tailored to peculiarities of individual systems. This paper presents an approach to mission-critical legacy system migration : the Butterfly Methodology, its data migration engine and supporting tool-kit framework. Data migration is the primary focus of the Butterfly methodology, however, it is placed in the overall context of a complete legacy system migration. The fundamental premise of the Butterfly methodology is to question the need for parallel operation of the legacy and target systems during migration. Much of the complexity of the current migration methodologies is eliminated by removing this interoperation assumption.

1. Introduction

The system engineering process normally involves phases of requirement definition, system design, sub-system development, system integration, installation, evolution and finally system decommissioning [22]. Within this process, system evolution plays, during the whole lifetime of the developed systems, a very important role in maintaining system performance and enhancing it to meet new requirements.

The widespread use of computer technology over several decades has resulted in some large, complex systems which have evolved to a state where they significantly resist further modification and evolution. Such systems are termed *Legacy Systems* [3]. These systems typically form the backbone of information flow within an organisation and are normally mission-critical : if one of these systems stops working the business will generally grind to a halt. Thus for many organisations, decommissioning these systems is not an option [22]. An alternative solution is legacy system migration which has become an important research and practical issue, for both the system engineering and database areas ([3], [22], [11] and [27]).

Legacy system migration is concerned with developing a target system which retains the functionality and, importantly, data of the original legacy system [3] but which can be easily maintained and adapted to meet future business requirements. There is an urgent need to provide a strategy which will allow the migration of the systems to new platforms and architectures and to provide tools and methodologies to support such a strategy.

In this paper an approach to legacy information systems migration is presented: the Butterfly methodology, and a framework tool-kit to support the methodology is outlined. In the following section current approaches to legacy system migration are briefly reviewed. Section 3 presents the Butterfly methodology in detail. Section 4 presents some detail of the Butterfly methodology for legacy data migration, a core part of a migration project. Section 5 then briefly discuss a framework of tool-kit for supporting the Butterfly methodology. Finally, Section 6 presents a summary of findings and discusses a number of future directions.

2. Related work

Legacy system migration encompasses many research areas. A single migration project could, quite legitimately, address the areas of reverse engineering, business reengineering, schema mapping and translation, data transformation, application development, human computer-interaction and testing. Due to space limitations, only a brief outline of research directly related to legacy system migration will be presented.

Current practical solutions mainly adopt what is referred to as "Wrapping". Wrapping involves surrounding existing data, individual programs, application systems, and interfaces to give a legacy system a 'new and improved' look or improve operations [26]. The most popular implementation of this is "Screen Scraping". Screen Scraping is the process of replacing the character based front ends of legacy systems with a client based graphical user interface. Introducing a graphical interface does not address many of the serious problems posed by legacy systems. At best it reduces training costs for new employees and allows an interface to the legacy system be placed on the desktop. Thus the problems legacy systems pose can only be overcome using a comprehensive migration strategy.

Tilley [27] discusses legacy system reengineering from several perspectives: engineering, system, software, managerial, evolutionary and maintenance. A framework for legacy system reengineering is proposed for each perspective. Using the system reengineering framework, the implication is that the legacy system will operate normally while the target system is developed independently. When the target system is complete, the legacy system will be shut down and the target system switched on. However, the proposed frameworks are presented at too high a level to be applied in practice and no consideration is given to the migration of legacy data. This is unacceptable when considering migration of mission-critical legacy systems.

Ganti and Brayman [11] propose general guidelines for migrating legacy systems to a distributed environment. Using these guidelines, the business is first examined and the business processes found are re-engineered as required. Legacy information systems are linked with these processes to determine which systems have data and business logic of value in the new target environment. A set of processes are selected and the associated legacy systems are analysed. New applications are then developed to fit these processes. These guidelines are not really suitable for use in migrating a mission-critical legacy system. Only the data and its structure are used in the decentralised target system. Mention is made of retaining logic encoded in applications but it appears that the legacy systems will be discarded and replaced with new applications. This approach recognises that legacy system migration should cause as little disruption to the current business environment as possible. However, it is unclear how the cut-over to the new, separately developed, target system will be handled.

In their *Chicken Little* Methodology Brodie and Stonebraker ([2], [3]) propose an 11 step generic migration strategy employing complex gateways, shown in Figure 1. In this method the legacy and target information systems are operated in parallel throughout the migration. Initially the target information system is very small, perhaps only one application with a very small database. However as the migration progresses the target system will grow in size until it performs all the functionality of the legacy system which can then be retired. During the migration, the legacy and target information systems interoperate to form the operational mission-critical information system. This interoperability is provided by a module known, in general, as a *gateway*, "a software module introduced between operation software components to mediate between them" [3].

Step 1 : Incrementally analyse the legacy information system
Step 2 : Incrementally decompose the legacy information system structure
Step 3 : Incrementally design the target interfaces
Step 4 : Incrementally design the target applications
Step 5 : Incrementally design the target database
Step 6 : Incrementally install the target environment
Step 7 : Incrementally create and install the necessary gateways
Step 8 : Incrementally migrate the legacy databases
Step 9 : Incrementally migrate the legacy applications
Step 10 : Incrementally migrate the legacy interfaces
Step 11 : Incrementally cut over to the target information system.

Figure 1 Chicken Little Migration Approach

An example of *Chicken Little's* general migration architecture is shown in Figure 2 (modified from [3]). Data is stored in both the migrating legacy and the growing target systems. A *forward gateway* is employed to enable the legacy applications access the database environment in the target side of the migration process and a *reverse gateway* is employed to enable target applications to access the legacy data management environment.

In most cases, gateway *co-ordinators* have to be introduced to maintain data consistency. However, as Brodie and Stonebraker themselves recognise, maintaining update consistency across heterogeneous information systems represents a complex technical problem which has no general solution and is an open research challenge [3]. Thus it seems that to apply the *Chicken Little* approach would represent a major challenge to any migration engineer.

In summary, the few complete migration methodologies available are either so general that they omit many of the specifics or are too complex to be applied in practice. Little focus is given to legacy data migration in most methodologies. Brodie and

Stonebraker's *Chicken Little* methodology offers the most mature approach. However, the need for the legacy and target systems to interoperate via gateways during the migration process adds greatly to the complexity of an already complex process and is also a considerable technical challenge. Thus a need exists for a safe, comprehensive, gateway-free approach to legacy system migration.

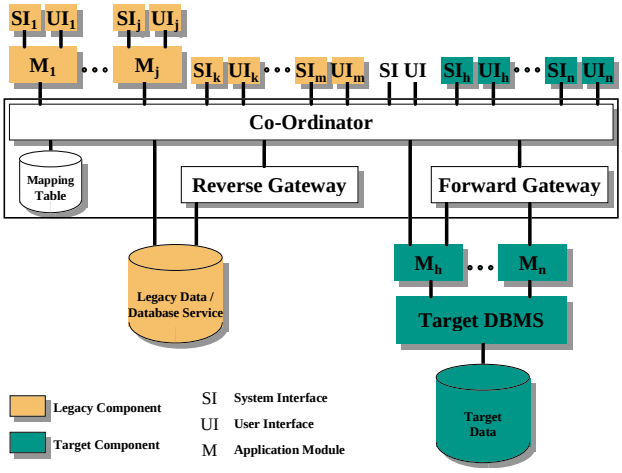

Figure 2 An Example of Chicken Little's General Migration Architecture

3. The Butterfly methodology

The Butterfly methodology is being developed as part of the MILESTONE project, a collaborative project involving Trinity College Dublin, Broadcom Éireann Research, Telecom Éireann, and Ericssons. MILESTONE aims to provide a migration methodology and a generic supporting tool-kit for the methodology to aid migration engineers in the process of migrating legacy information systems to target systems. The project began in July, 1996 and will finish in June, 1998. A trial legacy system migration following the Butterfly methodology is currently being planed and results will be available in the future.

The objective of the Butterfly methodology is to migrate a mission-critical legacy system to a target system. Different from *Chicken little*, the Butterfly methodology eliminates, during the migration, the need for system users to simultaneously access both the legacy and target systems, and therefore, eliminates the need of interoperation between these two (heterogeneous) information systems.

It is very important to bear in mind that, using the Butterfly methodology, **the target system will not be in production while the legacy system is being migrated**. The legacy system will remain in full production during the whole migration process. **There will never be a case where live data is stored, at the same time, in both the legacy and target systems.**

3.1 Overview

Legacy system migration can be a very expensive procedure which carries a definite risk of failure. In order to perform a successful migration, a sound model of the migration process is obviously needed. Currently, however, no general model exists.

MILESTONE's considers that migration consists of five major tasks:

1) Justification;
2) Legacy System Understanding;
3) Target System Development;
4) Migration
5) Testing.

This is illustrated in Figure 3.

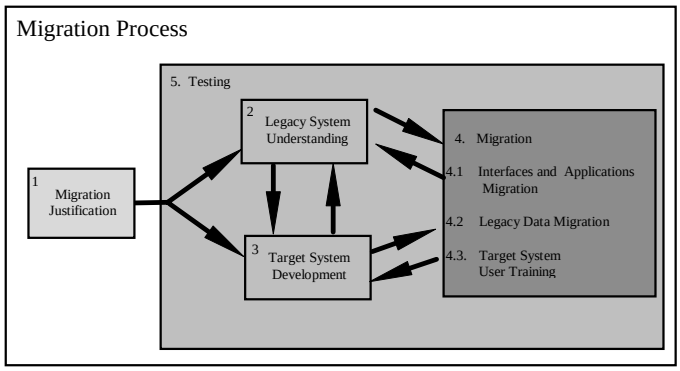

Figure 3 Major activities in legacy system migration

Within each task, general software/system engineering techniques can be applied. Testing plays a very important role in all tasks.

Justification involves an intensive study to quantify the risk and benefits and fully justify the redevelopment of the legacy system involved. Legacy System Understanding mainly involves *reverse engineering* of the legacy system. Target System Development is a constrained *forward engineering* task, the main constraint being the requirement for function-equivalence to the legacy system. Migration involves the physical transformation of whole the legacy system environment (i.e. legacy application, data, interface and the system users) to the target system. Due to its mission-critical nature, legacy system migration should cause as little disruption to the business environment as possible.

Within the process tasks are related and the process itself is iterative. Initially, *Justification* has to be done. Only after this, can *Legacy System Understanding*, *Target*

System Development and *Migration* be performed. *Testing* is addressed in all these tasks. Each task can be sub-divided into a number of sub-tasks. The results of these will affect the procedure and results of tasks in both previous and subsequent phases. In this sense, the migration process is a spiral model for tasks 2, 3, 4 and 5.

3.2 The Butterfly methodology phases

Using MILESTONE's view of the migration process, the Butterfly methodology divides legacy system migration into six major phases, Figure 4. An extra phase is introduced to produce sample data to facilitate testing and training. Activities related to data migration are presented in more detail because this is the particular focus of the Butterfly methodology.

Phase 0: Prepare for migration.
Phase 1: Understand the semantics of the legacy system and develop the target data schema(s).
Phase 2: Build up a Sample Datastore, based upon the Target SampleData, in the target system
Phase 3: migrate all the components (except for data) of the legacy system to the target architecture.
Phase 4: Gradually migrate the legacy data into the target system and train users in target system
Phase 5: Cut-over to the completed target system.

Figure 4 Six Phases of the Butterfly methodology

Before explaining the detail of the methodology, it is worth emphasising two points. **First,** the Butterfly methodology deliberately stores live data at the legacy system side during migration, and the target system will not be in production before the full migration process finish. This is different from gateway-based migration approaches where live data is distributed at both legacy and target systems during migration. This presents a great technical challenge to maintain data consistency, for which no general solution is available currently. **Second**, the Butterfly methodology proposes a legacy data migration engine, suitable for mission-critical system migration, so that the legacy system need only be shut down for a minimal amount of time. This differs from the so called *Big-Bang, Forward and Reverse Migration* [3] approaches where the legacy system must be shut down for a considerable time to facilitate data migration before the target system is made available.

- **Phase 0:** Prepare for migration.

Once the decision to migrate a legacy system has been made, the next stage is to prepare everything for the migration. Although many issues essential to a migration project must be clarified at this stage, for example management, organisational issues or budget, the Butterfly methodology focuses only on the technical issues. Among these issues the Butterfly methodology considers user requirements for migration and target system determination to be most important.

The main activities within this phase are listed in Figure 5. Obviously these activities can only succeed through intensive co-operation among the legacy system experts, migration engineers and users.

Phase 0: *Prepare for migration*
0.1 Getting the migration preliminary requirements;
0.1.1Determining user requirements;
0.1.2 Determining benchmarks for measurement of migration success;
0.2 Determining the target architecture;
0.3 Preparing the target hardware system;

Figure 5 Migration Activities in Phase 0

- **Phase 1**: Understand the semantics of the legacy system and develop the target data schema(s).

The activities identified within this "reverse engineering" phase are listed in Figure 6. Activity 1.4 is optional because a legacy system may not interact with other systems. Activity 1.5 to finalise the migration requirements is needed as it may not be possible to identify all the requirements until the legacy system has been understood.

A wide range of tools has been developed to assist in this reverse engineering area and there are more to come in future. The Butterfly methodology will take advantage of these tools and develop new tools only if it becomes absolutely necessary. One such tool is Data-Access-Allocator (*DAA*) developed by activity 1.6 which will be used to redirect all manipulations of legacy data, and data stored in TempStores once the data migration has begun, by the legacy system. (Refer to Section 4.)

Phase 1: *Understand the semantics of the legacy system and develop the target data schema(s)*
1.1 Understanding the legacy interfaces, identifying redundancies and determining the function of the target interfaces;
1.2 Understanding the legacy applications, identifying redundancies and determining the function of the target applications;
1.3 Understanding the legacy data; identifying redundancies and determining to-be-migrated data;
(optional) 1.4 Identifying and understanding interactions with other systems;
1.5 Finalising the migration requirements;
1.6 Developing the Data-Access-Allocator (DAA);
1.7 Developing the target data schemas and determining the mapping rules.

Figure 6 Migration Activities in Phase 1

- **Phase 2**: Build up a Sample Datastore, based upon the Target SampleData, in the target system.

The main activities of this phase are to determine the legacy SampleData and to develop the data transformer : *Chrysaliser*, another special tool required by the Butterfly methodology. The main function of Chrysaliser is to transform the legacy data, as well as the data in TempStores, to the target system. Initially, the legacy SampleData will be transformed by Chrysaliser to form the target Sample DataStore. This will be used to develop and test the target system. Figure 7 lists the activities involved in this phase. (Also refer to Section 4.)

Phase 2: *Build up a Sample Datastore, based upon the Target SampleData, in the target system*
- 2.1 Determining the Legacy SampleData;
- 2.2 Developing Chrysaliser;
- 2.3 Transforming the Legacy SampleData into the Target SampleData and building up the Sample DataStore;

Figure 7 Migration Activities in Phase 2

- **Phase 3:** Migrate all the components (except for data) of the legacy system to the target architecture.

"Forward" system engineering principles and methods will be one guideline for migration in this phase. The Sample DataStore, built up in Phase 2, will be used to support the cycle of the 'design-develop-test' for newly developed target components. Figure 8 lists activities in this phase.

Phase 3: *migrate all the components (except for data) of the legacy system to the target architecture*
- 3.1 Incrementally migrating legacy interfaces
  - 3.1.1 Migrating/developing a fragment of target interface;
  - 3.1.2 Testing it against Sample DataStore on Correctness;
  - *(optional)*3.1.3 Validating it against User's requirements;
- 3.2 Incrementally migrating legacy interface applications
  - 3.2.1 Migrating/developing a target application;
  - 3.2.2 Testing it against Sample DataStore on Correctness;
  - *(optional)* 3.2.3 Validating it against User's requirements;
- 3.3 Incrementally migrating reusable legacy components
- 3.4 Target components/system integration;
- 3.5 Target components/system testing on Correctness;
- 3.6 Target components/system validating against User's requirements;
- *(optional)* 3.7 Training users on target components/system;

Figure 8 Migration Activities in Phase 3

Activity 3.4, target components/system integration, is applied to test interactions among the components of the evolving target system. Because the evolving target system is built upon the Sample DataStore, some aspects of user training can be introduced here without the risk of data inconsistency. This activity is optional.

- **Phase 4:** Gradually migrate the legacy data into the target system and train users in target system.

This phase is mainly devoted to legacy data migration and is the core part of the Butterfly methodology. The legacy data will be gradually migrated into the target system by introducing a series of TempStores, the Data-Access-Allocator (DAA) and the data-transformer (Chrysaliser). The activities of this phase are listed in Figure 9. Section 4 will present an overview of data migration in the Butterfly methodology. One point may worth emphasising is that, during whole the data migration, legacy system always remains in full production.

Phase 4: *Gradually migrate the legacy data into the target system and train users in target system*
- 4.1 Incorporate DAA into legacy system;
- 4.2 Create TempStore TS_1; then set legacy datastore (TS_0) to read-only;
- 4.3 Migrate all the data in TS_0 into the target datastores through the *Chrysaliser*. While this migration is taking place all access to the legacy data store is redirected by the *DAA* and all the results of manipulations are stored into the new TempStore, TS_1. Continue until TS_0 has been fully migrated;
- 4.4 Create TempStore TS_2; then set TS_1 to read-only;
- 4.5 Migrate all the data in TS_1 into the target datastore(s) through *Chrysaliser*. As before all access to the legacy data is redirected by the *DAA* and all manipulation results are stored in the new TempStore TS_2. Continue until TS_1 has been fully migrated;
- 4.6 Repeat step 4.4 and 4.5 for TS_{n+1} and TS_n until the *Termination-Condition* is met, i.e. TS_n has been fully transformed and, at the same time, there exists $size(TS_{n+1}) \leq \varepsilon$;
- 4.7 Freeze the entire legacy system and migrate all the data in TS_{n+1} into target datastore(s) through the *Chrysaliser;*
- 4.8 Train users for the target system;

Figure 9 Migration Activities in Phase 4

- **Phase 5**: Cut-over to the completed target system.

The last phase of the Butterfly methodology is the cut-over phase, Figure 10. Once the target system has been built up and all the legacy data have been migrated, the new system is then ready to run.

Phase 5: *Cut-over to the completed target system*
- 5.1 Cut over to the completed target system.

Figure 10 Migration Activities in Phase 5

4. The Butterfly methodology data transformation

It is commonly agreed that the successful migration of the data management service from the legacy to the target system is the key to overcoming many of the problems posed by legacy information system [3]. The Butterfly methodology gives particular focus to the migration of legacy data in a mission-critical environment.

As can be seen from previous section, phases 2 and 3 of the Butterfly methodology deal with legacy system understanding and target system development. Target system development is supported by a sample-datastore, derived from legacy data and mapped to target side. Once phases 0, 1, 2 and 3 have finished, Phase 4 : data migration can then start. **Only after all data in the legacy datastore and TempStores has been transformed to the target system, will the target system be in production.**

Once data migration commences, the legacy data store is "frozen" to be read-only. Manipulations of legacy data are redirected by the Data-Access-Allocator (DAA), Figure 11, and the results stored to a series of auxiliary datastores named TempStore(s) (TS). When legacy applications access data, the DAA retrieves data from the correct source, e.g. the legacy data or the correct TempStore.

A Data-Transformer, named Chrysaliser, is employed to migrate, in turn, the data in the legacy system as well as in the series of TempStores to the target system. When Chrysaliser is migrating the legacy data (TS_0) all manipulations are stored in TS_1; when migrating the data in TS_1, all manipulations are stored in TS_2; and so on. In general, when data in TS_{n-1} has all been transformed into target system, then 1) TS_{n+1} is built up (with both read and write access right to DAA) to store the manipulation results of legacy application. 2) TS_n is "frozen" to be read-only and Chrysaliser migrates the data held in it.

The Butterfly methodology introduces a Termination-Condition, and a Threshold Value (represented by ε), to determine when migration has reached the requisite stage to cut-over to the target system. ε is a pre-determined[1] value representing the allowable amount of data in the final TS. If size(TS_n) ≤ ε, the amount of time necessary to migrate the data is sufficiently small to allow the legacy system to be brought down without causing any serious inconvenience to the core business. Thus using the Butterfly methodology, at no time during the migration process will the legacy system be inaccessible for a significant amount of time.

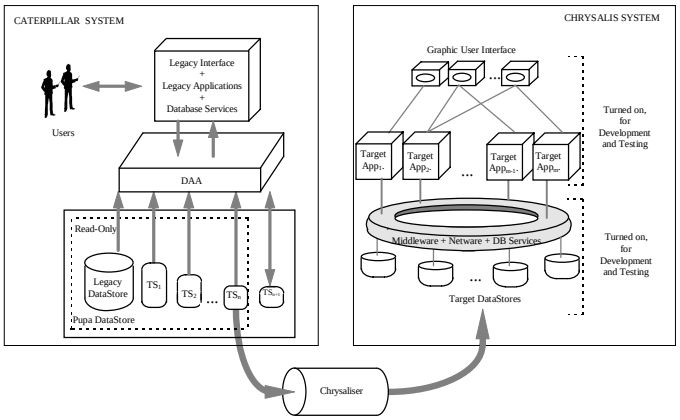

Figure 11 The Butterfly methodology, Migrating the Data in TempStore TS_n

Figure 11 shows a scenario during data migration. It can be seen that the DAA and Chrysaliser combine to serve as a data migration engine for the legacy data migration.

The legacy system can continue to operate normally during migration until the last TempStore has reached the pre-determined threshold value. At no time does the legacy system need to co-operate, during migration, with the target system.

Further discussion on the soundness and completeness, and the implementation mechanisms of DAA and Chrysaliser for different legacy system data models is beyond the scope of this paper. Details can be referenced to at ([29], [30], [31]).

5. The Butterfly methodology outline supporting tool-kit

Research into software tools has been ongoing for decades. Numerous tools have already been developed to assist in many stages of migration and research is still ongoing ([1], [4], [5], [6], [7], [12], [15], [17], [20], [21], [28]). However no single tool can completely automate any single phase, let alone migration as a whole.

The aim of MILESTONE is not to develop an integrated tool-kit for migration as a whole. Due to time and resource limitations, MILESTONE's main focus is restricted to identifying the requirements of supporting tools, with particular reference to the Butterfly methodology. Where possible, MILESTONE will take advantage of results from other research projects (or ideally off-the-shelf products) which serve its purpose, and only develop tools unique to the Butterfly methodology : DAA and Chrysaliser.

[1] Most likely by the Administrator of the Legacy System.

The following sections identify the supporting tools for the Butterfly methodology based on MILESTONE's view of the migration process. MILESTONE's tool-kit is intended to *support* rather than automate the migration process. The basic structure and flow of information in the MILESTONE tool-kit is illustrated in Figure 12. The following sections briefly outline tasks for which tools could be used and identify tools which could possibly be employed by the Butterfly methodology.

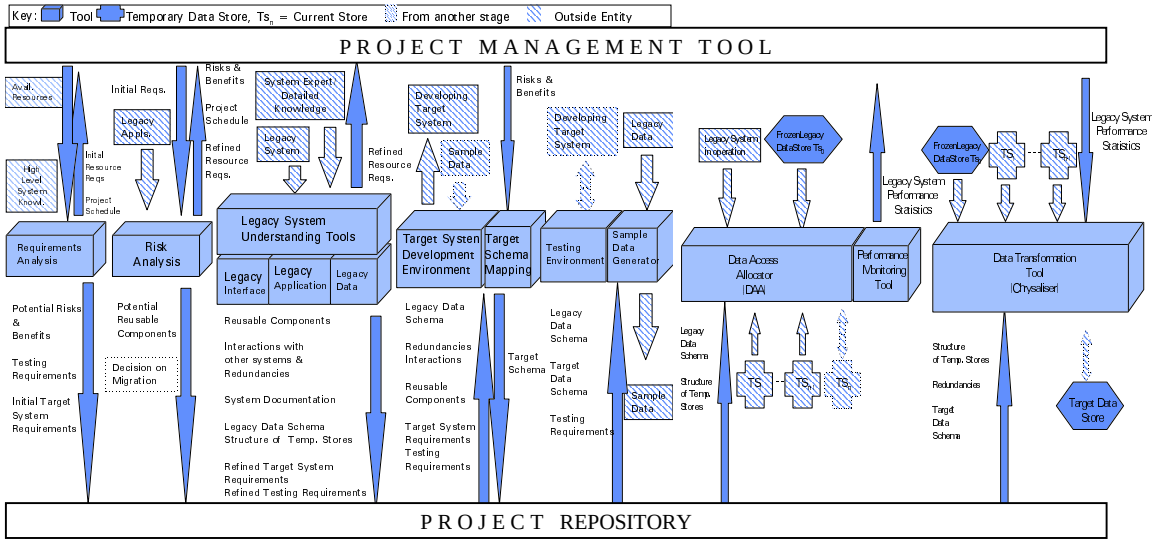

Figure 12 MILESTONE tool-kit framework and information flow

5.1 Project support

As for any large systems development project, migration using the Butterfly methodology will require the support of tools to manage the project schedule and resources. A conventional project management tool (e.g. [10], [14], [20]) may be used to support the management of the migration. A project repository will be used to store information needed to support tools used, and people involved, in all phases of migration.

5.2 Justification tools

The requirement for justification tools in migration is also similar to that for any systems development project. It is however more crucial that the risks and benefits of a migration be clearly understood as a failed migration project can result in an unusable legacy and an unusable target system. Obviously estimation of risk and benefit involves a degree of intuition based on experience, and knowledge of the individual organisation, and cannot be completely automated. However, once a plan has been formulated, cost and schedule uncertainty can be estimated using a risk analysis tool (e.g. [5], [6], [20]).

5.3 Legacy system understanding tools

Legacy system understanding is a core part of migration. Information retrieved or produced in this phase affects tasks in all other phases. Tasks to be performed include reconstructing system documentation, identifying and extracting the legacy data schema, identifying reusable components and redundancies and investigating how the legacy system interacts with other systems and resources. Much research has focussed on this area and numerous tools exist to assist in this process, although many may not be sufficiently mature for use in real migration efforts (e.g. [1], [4], [7], [12], [15], [17], [21], [28]).

5.4 Target system development tools

Any appropriate system development tools can be used to support the target system development. Numerous tools exist to assist in this task (e.g. [8], [13], [18], [25], [24]). There are also some tasks specific only to migration, such as schema mapping, for which tool support would be invaluable. Mapping tools will therefore be used to build the new target data schema from the legacy data schema (e.g. [7], [9], [16], [19], [23]).

5.5 Testing tools

Testing is an essential part of migration using the Butterfly methodology and is an ongoing process in all phases. Apart from the general testing environment needed for any system development, some particular requirements exist for migration testing. One important aspect of migration testing is to ensure that there are no unexpected inconsistencies between the critical functionality of the legacy system and its replacement. Unfortunately, to date, no specific migration testing environment exists. It is beyond the main focus of MILESTONE to develop such an environment. However, because it is important to use "legacy" data to test the target system i.e. actual data which will be used in the target system, an application/tool for sample data generation will be introduced in the MILESTONE tool-kit.

5.6 Migration tools

The critical part of a migration project is the cut over from the legacy system to the target system. An important migration requirement is to cause as little disruption to the business environment as possible. The most essential and difficult migration process is that of migrating the mission-critical legacy data. Using the Butterfly methodology, two tools are used to control this process : Data Access Allocator (DAA) and a Data Transformation Tool (Chrysaliser). These tools will be application specific and may have to individually constructed for each migration effort.

6. Conclusions and future work

In this paper MILESTONE's approach to the problem of legacy system migration has been presented. The migration process as a whole is a very complex procedure encompassing many different fields of research. The focus of discussion was thus necessarily limited. The proposed Butterfly methodology applies to the whole process of legacy system migration with the main focus specifically on the migration of legacy data in a mission-critical environment. The Butterfly methodology is a simple, safe, and open approach to this problem. It represents a departure from current thinking on how legacy systems as a whole can be migrated to new architectures.

The main difference between the Butterfly methodology and other proposed migration methodologies, is that the Butterfly methodology is a gateway-free approach. It eliminates, during migration, the need for system users to simultaneously access both the legacy and target systems. There is therefore, no need to keep consistency between these two (heterogeneous) information systems as the Butterfly methodology always stores live data at legacy system side!. In practice, using gateway-based approaches, gateway co-ordinators [3] have to be introduced to maintain data consistency. However, as Brodie and Stonebraker point out "Update consistency across heterogeneous information systems is a much more complex technical problem with no general solution yet advised, and it is still an open research challenge" [3]. Thus to design and implement a gateway co-ordinator is a task without general methods or even guidelines. In contrast, generic mechanisms of the Butterfly methodology's data transformation engine : DAA and Chrysaliser for legacy (flat file, hierarchical) systems have been developed [31].

MILESTONE is an ongoing project, working with real life legacy systems in Telecom Éireann, the Irish national telecommunications service provider. Immediate future work of MILESTONE includes further investigating the framework tool-kit which supports the Butterfly methodology. Future work also includes an effort to implement the *Chrysaliser* Data Transformer and the *DAA* Data-Access-Allocator subsystems for a migration process, and criteria and techniques to develop a Sample Datastore. MILESTONE is aware that many factors such as the structure of the TempStores and the placement of the Chrysaliser and DAA will affect the migration process and is investigating these issues. A trial migration applying the Butterfly methodology to a legacy system is being planned, and the results will be available when the project ends.

7. References

[1] Bachmann, 'A CASE for Reverse Engineering', Datamation, pp. 49-56, July 1988

[2] M. Brodie and M. Stonebraker, 'DARWIN: On the Incremental Migration of Legacy Information Systems', Technical Report TR-022-10-92-165 GTE Labs Inc., http://info.gte.com/ftp/doc/tech-reports/tech-reports.html, March 1993

[3] M. Brodie and M. Stonebraker, 'Migrating Legacy Systems: Gateways, Interfaces and the Incremental Approach', Morgan Kaufmann 1995

[4] D. N. Chin and A. Quilici, 'DECODE: A Co-operative Program Understanding Environment', Journal of Software Maintenance 8(1), pp. 3-34, 1996.

[5] Concept Sales Ltd., 'Dependency Modelling Tool (DMT)', http://www.bucks.net/concept/dmthome.html, May 1997

[6] C/S Solutions Ltd., 'Risk+', http://cssolution.com/riskov.

htm, May 1997
[7] Computer Science Department - The University of Namur, 'DB-MAIN: A R&D Programme in Database Applications Engineering and Case Technology', http://www.info.fundp .ac.be/~dbm/, February 1996
[8] Dynamics Research Coporation, 'VisualMagic', http://www.visualmagic.com, May 1997
[9] Evolutionary Technologies International, 'ETI-EXTRACT Tool Suite', http://www.evtch.com, May 1997
[10] Fresnelsoft, 'ReFind', http://www.fresnelsoft.com, May 1997
[11] N. Ganti & W. Brayman, 'Transition of Legacy Systems to a Distributed Architecture', John Wiley, 1995
[12] Z-Y Liu, M. Ballantyne and L. Seward, 'An Assistant for Re-Engineering Legacy Systems', Proc. 6th Innovative Applications of Artificial Intelligence Conf. pp 95-102, Seattle, WA, http://www.spo.eds.com:80/eds/papers/ asstreeng.html, August 1994
[13] Magna Software Corporation, 'Magna X Application Generator', http://www.magna.com, May 1997
[14] Microsoft Corporation, 'Microsoft Project', http://www.microsoft.com/project, May 1997
[15] Dr. H. A. Muller, 'Understanding Software Systems Using Reverse Engineering Technologies Research & Practice', Tutorial presented at Int. Conf. on Software Engineering 18, http://tara.uvic.ca/UVicRevTut/UVicRevTut.html, March 1996
[16] B. Narasimhan, S. B. Navathe and S. Jayaraman, 'On Mapping ER and Relational models into OO schema', Proc. 12th Int'l Conf. on the entity-relationship Approach ER'93 (LNCS 823) pp. 402-413
[17] J. Q. Ning, A. Engberts and W. Kozaczynski, 'Automatic Support for Legacy Code Understanding', In Communications of the ACM, 37(5), pp. 50-57, May 1994
[18] Object Domain Systems, 'Object Domain Object Oriented Analysis and Design Tool', http://www.object-domain. com, May 1997
[19] Ontos Inc., 'OIS Development Tools', http://www.ontos.com, May 1997
[20] Quantitative Software Management(QSM), 'SLIM (Software Life-Cycle Model)',http://www.qsm.com, May 1997
[21] Dr. H. Muller, 'RIGI Project - An Extensible System for Retargetable Reverse Engineering', University of Victoria, Canada, http://tara.uvic.ca, November 1996
[22] I. Sommerville, Software Engineering Environments, 5thEd., Addison-Wesley, 1995.
[23] F. N. Springsteel, 'Object-based schema integration for heterogeneous databases : a logical approach', Proc. 4th Int'l Conf. of database and expert systems Applications DEXA 93 (LNCS 720) pp. 166-180
[24] Structured Technology Group, 'AxiomSys, AxiomDsn', http://www.stgcase.com, May 1997
[25] S. Stobart, 'CASE Tool Home Page', http://www.osiris. sunderland.ac.uk/sst/casehome.html, May 1997
[26] Systems Techniques Inc., 'Wrapping Legacy Systems for Reuse : Repackaging vs. Rebuilding', http://www.systecinc.com/white/wplist.html
[27] S. R. Tilley and D. B. Smith, 'Perspectives on Legacy System Reengineering', http://www.sei.cmu.edu/ ~reengineering/lsyree
[28] K. Wong, S. Tilley, H. Muller, M. Storey, 'Structural Redocumentation: A Case Study', IEEE Software, pp. 46-53, January 1995.
[29] B. Wu, D. Lawless, J. Bisbal, J. Grimson, V. Wade, R. Richardson and D. O'Sullivan, 'The Butterfly Methodology : A Gateway-free Approach for Migrating Legacy Information Systems', Proc. Of 3rd IEEE Conf. on Engineering of Complex Computer Systems (ICECCS97), Como, Italy. Sept. 1997.
[30] B. Wu, D. Lawless, J. Bisbal, J. Grimson, V. Wade, R. Richardson and D O'Sullivan, 'Legacy Systems Migration : A Legacy Data Migration Engine', Proc. 17th Int'l Database Conf. (DATASEM'97), Brno, Czech Republic,Oct.1997.
[31] B. Wu, J. Bisbal, D. Lawless, J. Grimson, V. Wade, R. Richardson and D O'Sullivan, 'Implementation Mechanisms of DAA and Chrysaliser for Hierarchical Database and Flat file Systems', Technical Report, Trinity College Dublin, March 1997.