SOFTWARE RE-ENGINEERING

SE-409

Legacy System

- A legacy system is an old and outdated computer system, software application, programming language, or hardware device that is still being used by an organization despite being superseded by newer technology.
- These systems may still be functional, but they may not be as efficient, secure, or reliable as their newer counterparts. Also, they often require significant maintenance and support, which can be costly and time-consuming.
- Legacy systems are often difficult and expensive to maintain and upgrade, and they may be incompatible with modern hardware and software.

Why legacy systems are still be used?

- Legacy systems are still used by organizations for a variety of reasons:
  - One reason is that these systems may contain critical data or proprietary software that cannot be easily transferred to a new system.
  - Replacing a legacy system can be very costly and disruptive, and organizations may not have the resources or expertise to undertake such a project.
  - Some legacy systems may contain unique features or functionality that cannot be replicated in newer systems.
  - Many organizations continue to use legacy systems because they have been in use for many years and are familiar to staff, making them easier to operate and maintain.

Solutions for legacy information system (LIS)

There are several categories of solutions for legacy information system (LIS):

- **Freeze:** The organization decides no further work on the legacy system should be performed.
- **Outsource:** An organization may decide that supporting software is not core business, and may outsource it to a specialist organization offering this service.
- **Carry on Maintenance:** Despite all the problems of support, the organization decides to carry on maintenance for another period.

Solutions for legacy information system (LIS)

- **Discard and redevelop:** Throw all the software away and redevelop the application once again from scratch.
- **Wrap:** It is a black-box modernization technique that surrounds the legacy system with a software layer that hides the unwanted complexity of the existing data, individual programs, application systems, and interfaces with the new interfaces.
- **Migrate:** Legacy system migration basically moves an existing, operational system to a new platform, retaining the legacy system’s functionality and causing minimal disruption to the existing operational business environment as possible.

WRAPPING

- In 1988, Dietrich, first introduced the concept of a “wrapper” at IBM.
- Wrapping is a straightforward way to modify and enhance a legacy component.
- Wrapping means encapsulating the legacy component with a new software layer that provides a new interface and hides the complexity of the old component.
- The goal of this technique is to hide the complexities of the legacy system, making it easier to interface with newer technologies and systems.
- The encapsulation layer can communicate with the legacy component through sockets, remote procedure calls (RPCs), or predefined application program interfaces (API).

WRAPPING

Two important concept of wrapping:

- **Software Layer:** The "wrapping" involves creating a new software layer around the legacy system. This layer acts as an intermediary or interface between the legacy system and the modern components.
- **Complexity Hiding:** The primary objective of wrapping is to hide the unwanted complexity of the legacy system. This complexity may include outdated code, non-standard data structures, or dependencies that are not compatible with modern technologies.

Types of Wrapping

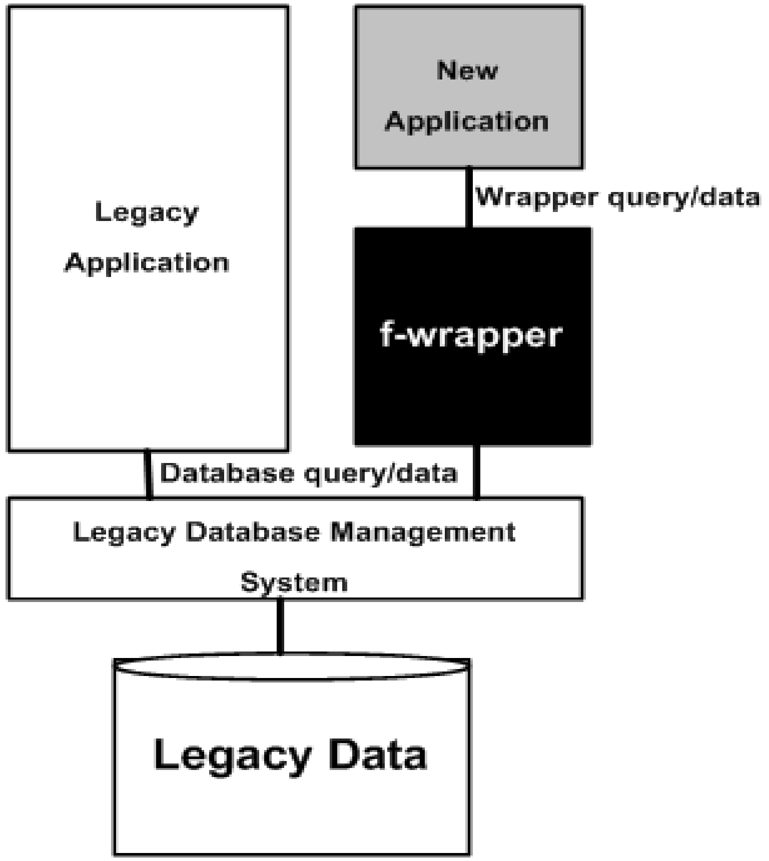
- Orfali et a. classified wrappers into four categories:
  - Database wrappers,
  - System service wrappers,
  - Application wrappers &
  - Function wrappers.

Database wrappers:

- Database wrappers can be further classified into forward wrappers (f-wrappers) and backward wrappers (b-wrappers).

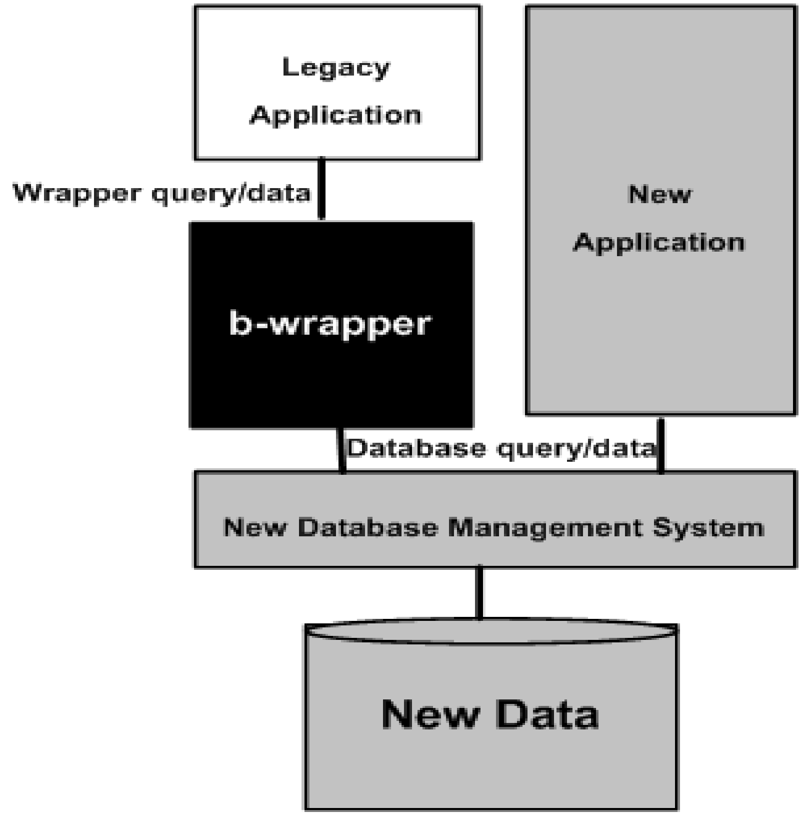
Forward Wrapper

- The forward wrappers-approach, depicted in Figure, shows the process of adding a new component to a legacy system.
- Functionality: When an application using modern technologies sends a request to interact with the database, a forward wrapper converts or maps that request into a form that is compatible with the legacy database's structure and syntax.



Backward Wrapper

- The backward wrappers-approach has been depicted in previous Figure, Backward wrappers handle the translation or transformation of responses from the legacy database into a format that is understandable by modern applications.
- Functionality: When the legacy database returns data in its native format, a backward wrapper converts or maps that data into a format that aligns with the expectations of the modern application.



Types of Wrapping

System service wrappers:

- **Definition:** System service wrappers encapsulate system-level functionalities or services, providing an intermediary layer between the application and the underlying operating system services.
- **Purpose:** These wrappers can be used to enhance security, manage resources, or provide additional functionalities such as logging, monitoring, or access control. They abstract the complexities of system-level interactions, allowing applications to interact with the operating system in a more controlled and secure manner.

Application wrappers:

- **Definition:** Application wrappers encapsulate entire applications or components, providing a standardized interface or protocol for communication and interaction with other applications or systems.

Types of Wrapping

- **Purpose:** Application wrappers facilitate integration, interoperability, and reuse of application components. They can be used to adapt legacy applications to modern environments, enable cross-platform compatibility, or enhance the functionality of existing applications by adding new features or capabilities.

Function wrappers:

- **Definition:** This kind of wrappers provide an interface to call functions in a wrapped entity.
- **Purpose:** Function wrappers are commonly used for implementing aspects such as logging, error handling, security checks, or performance monitoring.

Constructing a Wrapper

- Constructing a wrapper for a legacy system involves creating a new layer of code that sits between the legacy system and other software applications or systems.
- The purpose of the wrapper is to provide an interface that abstracts the functionality of the legacy system and makes it accessible to other systems without exposing its underlying complexity or limitations.
- **Some steps to follow when constructing a wrapper for a legacy system:**
  - Identify the functionality of the legacy system that needs to be exposed or integrated with other systems.
  - Choose an appropriate programming language and development framework for the wrapper.

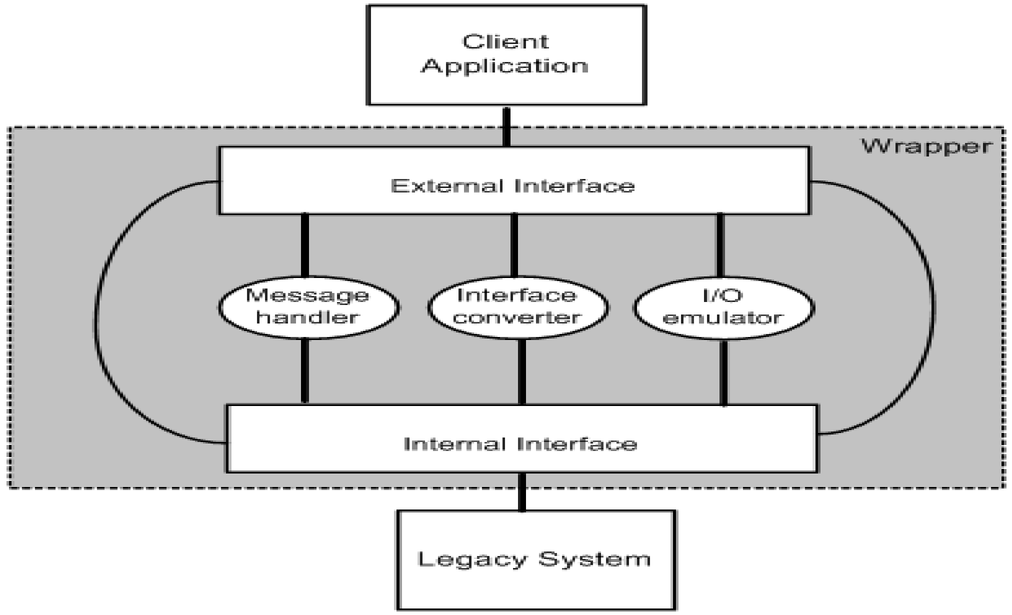
Constructing a Wrapper

- Design the wrapper interface to reflect the needs of the users and the capabilities of the legacy system. This can involve defining input and output parameters, error handling, security, and other features.
- Implement the wrapper code, using the appropriate programming techniques and best practices.
- Test the wrapper thoroughly, using a range of input data and use cases. This can involve unit testing, integration testing, and user acceptance testing.
- Deploy the wrapper in the target environment, and monitor its performance and usage. This can involve logging and tracking usage metrics, and providing support and maintenance to users and stakeholders.

Overall, constructing a wrapper for a legacy system requires a combination of technical skills, domain knowledge, and project management expertise

Constructing a Wrapper

- Conceptually, a wrapper comprises two interfaces and three event driven modules as shown in Figure.
- The two interfaces are an internal interface and an external interface, and the three modules are message handler, interface converter, and I/O-emulator.



Modules of a wrapping framework

Migration

- Migration in a legacy system refers to the process of moving an existing software application or system from an old or outdated platform or technology to a newer one.
- This is done to improve the performance, security, and functionality of the system, and to ensure its compatibility with the latest hardware and software environments.
- Migration can involve transferring the entire system to a new platform, or only certain components, data, or functionalities.
- A migration project can be complex and risky, as it requires careful planning, testing, and validation to ensure that the new system works as expected and meets the requirements of the users and stakeholders.

MIGRATION METHODS

- There are several methods for migrating from a legacy system to a newer platform or technology.
- Big Bang Approach: This involves completely replacing the legacy system with a new system in one go. It is a high-risk approach as the new system has to be fully developed and tested before the migration.
- Phased Approach: This involves breaking the migration into smaller, manageable phases. Each phase is implemented and tested, and then rolled out to users. This approach is less risky than the big bang approach.
- Parallel Run Approach: In this approach, the new system is run in parallel with the legacy system for a period of time. This allows users to become familiar with the new system while still using the old system. Once users are comfortable with the new system, the legacy system is switched off.

MIGRATION METHODS

- Pilot Approach: This involves implementing the new system in a small, controlled environment, such as a single department or location, before rolling it out to the entire organization.
- Hybrid Approach: This approach involves combining several of the above methods to suit the specific needs of the migration project. For example, a phased approach may be used for some parts of the migration, while a parallel run approach may be used for others.

Software/Data Migration tools

- **Redgate Flyway:** Flyway, by Redgate, automates database deployments across teams and technologies.
- IBM Lift: IBM Lift is a tool provided to migrate on premise databases to IBM Cloud.
- Quest Migration Manager: Quest offers Migration Manager, replacing Dell's former ChangeBASE migration software, a suite of application tools supporting Microsoft application and legacy system migration.
- AWS Database Migration Service: AWS Database Migration Service helps users migrate databases to AWS. The source database remains fully operational during the migration to minimize downtime to applications that rely on the database.