Entity Life Cycle Testing

Manunandan Subrahmanyan
Mohan Kumar K.L.
Banking Business Unit
Infosys Technologies Limited, Bangalore

Abstract

Entity Life Cycle Testing is employed during System Testing where validation is done for an entity which undergone several processes' before it reaches the current Scenario under focus. How this differs from Business Scenario Testing is that a Scenario Test Case talks about the end to end of particular cycle of the Entity, whereas Entity Life Cycle Testing talks about all the Scenarios' the entity Traveled/Undergone during its life, before it reaches the current Scenario.

This kind of Testing is more suitable for multi-complex applications like Banking and Insurance where the financial implications' are wider in most of the stages, if not in all the stages. This testing not only validates the current cycle also captures regression, if any, after Integration Testing is done.

What is Entity Life Cycle Testing

An entity in application software is the least granular object that can be identified. For Example an Account or a Customer Id is an entity in a Banking Application. In a financial application, it is very important to identify the entities involved and clearly define the path each entity is going to traverse during its life cycle. Different business scenarios could evolve at each life cycle stage for the entities.

Defining the life cycle stages, identifying and mapping business scenarios at each life cycle stage and tracking the entity through the entire life cycle forms the crux of Entity Life Cycle Testing.

This is the most effective technique of testing complex financial applications like Banking or Insurance during System Testing or Regression Testing.

What are the challenges in regression testing of a large and complex financial package?

The most daunting task in undertaking the testing of a large and complex financial package is the sheer magnitude of test cases and maze of business scenarios that have to be tested. The inter-dependencies of different business life cycles have to be clearly understood and paths have to be drawn to ensure that all the paths have been covered. Data accumulation/redundancy, if any, in all the life stages of the Entity are to be checked so that wrong updates/no updates should not result in wrong results in other operations or MIS reports. This is more so when an existing application is migrated into a new system or upgraded into a higher version.

The conventional methods of System Testing may not cover all the processes/paths covered by an entity during its life cycle in their entirity.

How entity life cycle testing simplifies this problem

In entity life cycle testing, the testing flow is driven by the life cycle stages of a business entity. At each life cycle stages, the different paths that the entity could take are clearly identified and test cases derived to cover all the paths. The path intersection of different entities is hence covered automatically into the scheme of testing, thereby ensuring that the uncovered paths are totally eliminated.

At each life cycle stage, there could be a single or multiple business scenarios emerging. Thus, each business scenario is covered end-to-end at each life cycle stages.

What are the steps involved

The steps involved in Entity Life Cycle Testing are:

- Identification of Business entities
- Identification of Life Cycle Stages
- Identification of Business Scenarios at each Life Cycle Stage
- Mapping of Business Scenarios for cross-functional dependencies and designing test cases
- Executing the Life Cycle Testing by taking the entity through the life cycle stages by pre-defined stages

Entity Life Cycle Testing - A case study:

This method was applied to Finacle™ during the System Testing and Migration Testing.

When Finacle was launched, there arose a need to migrate many existing users of Bancs2000 to Finacle. The existing techniques of System Testing/Migration testing were found to be wanting in catching regression defects and serious functionality loss. Hence we had to devise a new technique that had a high probability of catching such defects before releasing the product to the customer.

Of the many alternatives considered, the Entity Life Cycle Testing method offered a great promise in catching all regression bugs that could crop up at different life cycle stages. Hence this method was adopted.

This paper will discuss the methodology used for system testing and regression testing using Entity Life Cycle Testing and the benefits derived from the process.

Methodology

The Finacle™ product was broken down into different modules and business entities were identified in each module. The Life cycle Stages of each entity were identified and business events/scenarios were mapped with entities at different Life Cycle Stages.

Data seeding was done in Bancs2000 to have at least one entity at each life cycle stage before migration. Database was migrated to Finacle™ by applying the migration scripts. Post migration, entity life cycle test cases were executed and the behaviour of the entities were observed at each subsequent life cycle and compared against the behaviour in Bancs2000.

This method proved to be very effective in catching the regression defects or lost functionalities after migration and also to identify data redundancy/accumulation, hence misinforming MIS reports.

The direct advantage of using this method is in reduction of acceptance testing defects from site as illustrated in the following table:

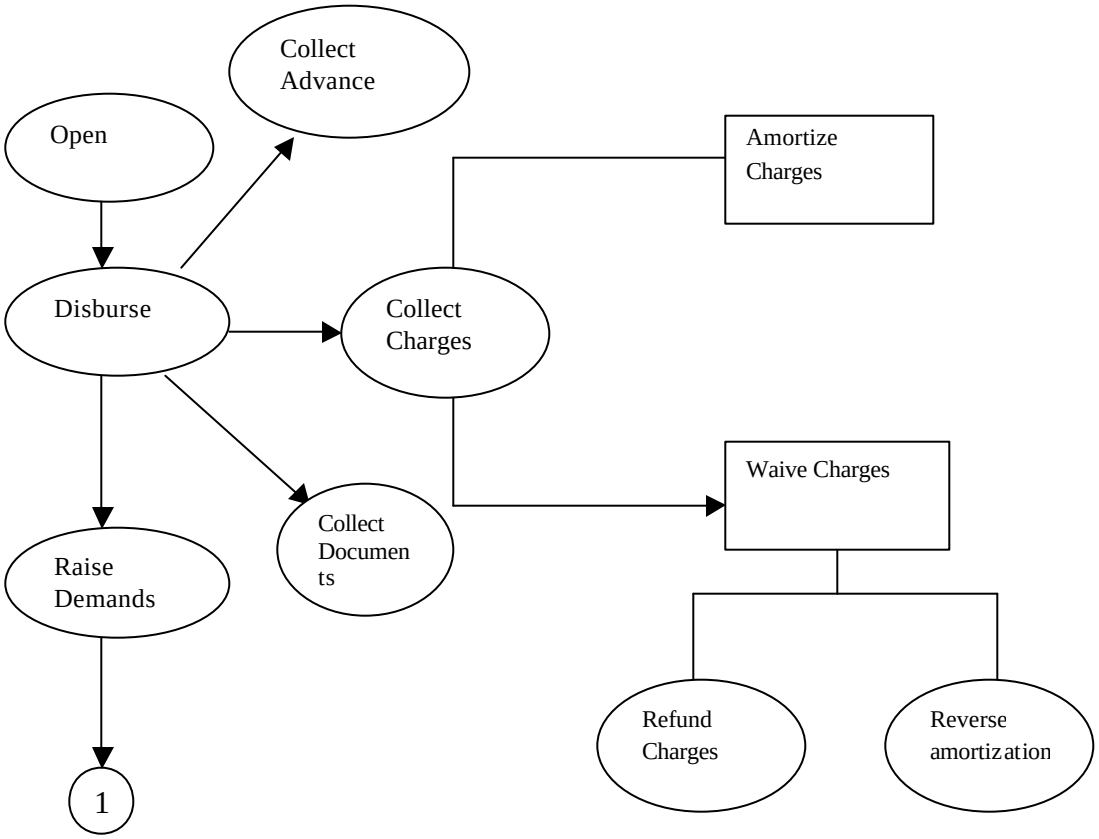
Release	Delivered defects / KLOC	Delivered defects/person month	Delivered defects/FP
PCB 2.0	1.28	1.35	0.16
PCB 3.0	1.28	1.04	0.13
PCB 4.0	1.17	0.97	0.12
PCB 5.0	1.09	0.99	0.11

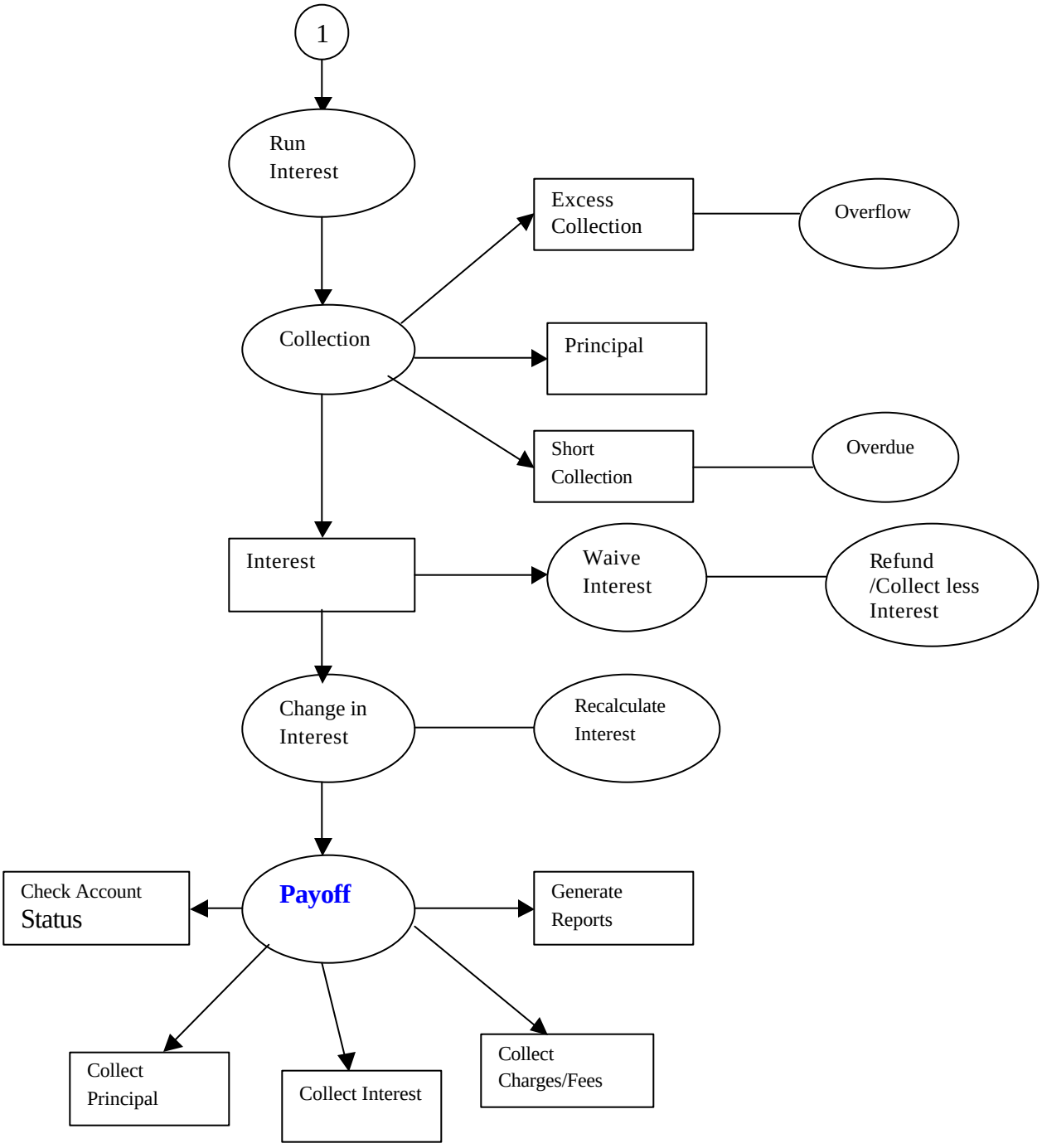
Other collateral advantages we derived by using this method are:

- Different Business Scenarios of different customers, which were hitherto undocumented, were detected - helps while deploying a release to the customer
- Better understanding of product steps for completing a business scenario
- Helped in continuous improvement of the product – by increasing the test coverage.
- Identification of commonly used scenarios - helps in finding the impact of changes being made to a module - Any defect found will have larger impact.
- Identification of different paths a business entity could take – helps in developing scripts for automation of regression testing.

An Example:

Payoff of a Loan Account Possible Entity Flows:





In this example the Scenario starts from Payoff process; where, Test cases are written to check for the Payoff process and the related events like Collection of Outstanding Principal, Interest, and Charges/Fees and also the MIS reports related to Payoff process.

Whereas in an Entity Life Cycle Testing model, Test Cases are written to check the result of all the process' involved during the every stage of the Entity (The Loan Account), during one of the final Stage/Scenario of the Entity (here the final process is Payoff/Pre-mature closure) after which the link between the Customer and the Bank comes to an end, as for as this Entity is concerned.

As this model covers' checking of each of the process involving an entity, this also captures defects , which are slipped out of Integration, and other Scenario testing, hence stops slipping out to Acceptance Test level.