

Optimizing Framework Release Plans Using the Incremental Funding Method

Andre de Souza Andrade^{1*}, Eber Assis Schmitz¹, Antonio Juarez Alencar¹, Alexandre Luis Correa²

¹ The Tercio Pacitti Institute for Computer Research, Federal University of Rio de Janeiro (UFRJ), Rio de Janeiro, Brazil.

² Department of Applied Informatics, State of Rio de Janeiro Federal University (UNIRIO), Rio de Janeiro, Brazil.

* Corresponding author. Tel.: +55 21 981359064; email: souza.andre@yahoo.com.br

Manuscript submitted October 5, 2015; accepted December 5, 2015.

doi: 10.17706/jsw.11.2.182-192

Abstract: This paper presents a method for the definition of the optimal release plan of framework features. The method is based on the economic benefits brought by the use of frameworks as platforms for the development of specific business applications. In the method, framework components (both frozen and hot spots) of new releases are considered to be Architectural Elements (AEs), whilst modules composing the business applications that customize the framework are considered to be Minimum Marketable Features (MMFs). The business applications teams provide the framework development team with the requirements for the features to be included in the next AE releases. While the AE team is focused on the development costs of AE releases, the business applications teams are concerned with the financial returns enabled by the new MMF deployment. The method provides a systematic approach to plan the implementation order of both framework and business application releases. This allows for a balanced view of the total financial value yielded by framework components and business applications.

Key words: Frameworks, incremental funding method, software engineering economics, software release Planning.

1. Introduction

Reuse of software assets (models, architecture and source code) together with a deep knowledge of the core business domain is a well-known strategy used by solution providers in order to promote IT-business alignment in organizations.

For the reason to effectively reuse software assets, a hypothetical software solution provider is organized in two types of development teams: *development for reuse team* which are compounded of system analysts and developers with a strong background in domain analysis and development of reusable artifacts such as frameworks; *development with reuse team*, who work closely to internal customers (stakeholders, users, decision makers) on the development of business applications.

Considering that the development for reuse staff usually has a backlog of several framework features that should be developed in order to be reused in the development of specific business applications, one problem that arises in this context is the definition of a release plan for the framework features. This involves defining the scope (features) and time frame for each framework release.

This paper presents a method for the definition of a release plan for features of a framework. The method is based on the Incremental Funding Method (IFM), which helps on the definition of an order of development based on the value generated by each business application module. Since a business application module may reuse features of a framework (when such a module is developed as a customization of the framework features), a release plan for the framework features is defined based on the economic benefits brought by new releases of a framework in the development of specific business modules.

The method for framework release planning proposed in this paper is applicable to environments where the framework development team and business applications development teams work in an integrated manner. This integration assumes that the framework development team can identify the set of framework features that will be reused by the business applications in the development of new systems.

The method also requires the availability of the cost estimations for the development of each framework feature and of the cost estimations for each business application module developed as a customization of one or more framework features. Estimations of the economic benefits (or revenues) brought by new business applications features are also required to the definition of an implementation order with a higher return on the investment.

The remainder of this paper is structured as follows: Sections 2 and 3 briefly describe relevant concepts used in the method, Section 4 describes the proposed method and some conclusions are drawn in Section 5.

2. Frameworks

A software framework is an architecture or infrastructure intended to support and enable the integration and interoperation of software components. It may consist of concepts, technologies, tools, protocols, standards, control mechanisms, interfaces, and processes intended to enable the rapid, efficient, and flexible assembly of systems from components in a practical setting [1].

Frameworks are a key concept in software reuse as they can bring many benefits: (1) less effort to build new applications, since several features have already been built and are ready to use; (2) higher quality applications because frameworks, as is usual in development for reuse efforts, should be developed following high quality standards and rigorous quality control procedures before being released; (3) higher abstraction for application development which means that developers can focus on developing specific business logic to be plugged into a robust infrastructure provided by the existent frameworks; (4) reduced time to market once the learning curve for a given framework has been overcome, because building a new application by customizing and extending a framework is much faster than developing it from scratch.

Adair [2] proposes a classification based on the aspect "scope". Johnson and Foote [3] define two types of frameworks, considering the aspect "customization". Andersen Consulting [4] differentiates the frameworks into two categories under the aspect "interaction". Table I summarizes those categorizations from the point of view of the three aspects mentioned above.

Table 1. Categorization of Frameworks

Aspect	Type	Description
SCOPE	Application frameworks	Support wide variety of domains
	Domain frameworks	Specific functionalities to a particular domain
	Support frameworks	Functionalities at the system level, from which other frameworks will be built
CUSTOMIZATION	White box frameworks	Architecture-oriented, based on inheritance
	Black box frameworks	Data-driven, based on composition (implementation of interfaces)
INTERACTION	Called frameworks	Applications use the framework by calling functions of the framework
	Calling frameworks	Applications provide business logic at specific points, called by the framework

3. Incremental Funding Method

3.1. Overview of the Method

The main idea of IFM is that the investment in a software development project should be incremental. This incremental investment intends to make the software development project self-funded, i.e., the project should afford the expenses and investment costs through its own financial resources arising from anticipated deliveries of software that create value for the business [5].

In order to effectively plan these early deliveries, it is necessary to identify the software application features, define their precedence relations and estimate the development cost and business value generated by each feature. To meet these requirements, it is crucial to strength the relationship between business experts, financial, marketing analysts and IT development team.

Many benefits can be achieved through IFM: increasing company profits through increased revenue generation or cost reduction, competitive differentiation, projection or consolidation of a brand, increased customer loyalty. Therefore, according to IFM principles, successful software projects are not simply those delivered on budget and within the schedule, but those that reach the organizational goals and bring significant financial return as soon as possible.

3.2. Minimum Marketable Feature (MMF) and Architectural Element (AE)

MMFs and AEs are the key building blocks of IFM. A MMF represents a partition of a software application or a software component which is considered by customers as important or essential to support the activities of an organization, thus generating business value.

MMF creates business value for an organization if, in line with the organizational strategic objectives, it includes at least one of the following critical success factors is on focus: competitive differentiation, increased sales (or increased revenue), cost reduction, brand projection and increased customer loyalty.

The architecture of a software system can be decomposed into smaller partitions called Architectural Element (AE). Unlike MMF, AE is considered as an infrastructure element that does not directly generate business value to the organization.

IFM uses sequencing strategies to identify the best moment to build each element of an application, i.e., AEs and MMFs. The goal is to develop each AE just before as they are needed to support the functionality of each MMF in order to optimize the financial return of the project.

Domain experts and business specialists should always participate in the decision regarding the classification of software parts as AEs and MMFs. In general, the architectural elements are related to services or features not directly perceived by end users.

Briefly, the IFM method has the following steps:

- 1) Group the features and functionalities of the software product into two categories: architectural elements (AE) and minimum marketable features (MMF).
- 2) Identify the precedence relationships among the features, i.e., identify constraints on the order in which AEs and MMFs may be developed.
- 3) Estimate the cash flow for each AE and MMF, considering the investment in the construction of the functionality and the value generated by it in each time slot.
- 4) Calculate the sequence-adjusted net present value (SANPV), which is the sum of the discounted cash flows of all time slots, for both AE and MMF, but adjusted according to the time slot when the development of AE/MMF starts.
- 5) Choose the release order of AEs and MMFs which will generate the maximum Return on Investment (ROI), considering the precedence constraints.

A practical example of the application of IFM can be found in Denne [6].

4. Optimal Framework Release Plans Using IFM

4.1. Proposed Method for Framework Release Planning

The input for the proposed method is the backlog of MMFs of the new business applications that should be developed and the backlog of all AEs corresponding to framework components, which will support the development of the new business applications.

Each AE corresponds to framework components (frozen and hot spots) that implement a new feature to be included in a future release of the framework. The requirements of each AE come from the needs of the framework users, namely, the development teams of business applications. The framework development team can then estimate the development cost of each feature.

Likewise, each MMF corresponds to the modules of the business applications that requires framework features previously developed, some of which to be implemented in the new release. These business application teams should be able to evaluate the ROI of the customizations enabled by the new feature.

The proposed method can be described in four steps:

- 1) Elicit new business applications features and identify cash flow (development costs and revenues obtained after deployment of the module) for each MMF business application.
- 2) Associate the new business features described in the previous step with new framework features and estimate the development cost of each feature, represented as an AE.
- 3) Determine the precedence relationships and constraints between MMFs and AEs.
- 4) Use of sequence-adjusted net present value (SANPV) heuristic to identify the optimal framework release plan for framework features, where each release has one or more features.

4.2. Running the Method

The method is demonstrated with a real world example based on a framework named *Evaluation Pro*, developed by an Oil&Gas company.

Evaluation Pro is a domain framework whose main characteristic is the use of questionnaires, with objective multiple-choice questions (applying automatic correction) and discursive subjective questions (supporting correction and assignment of score by the evaluator), including repository of questions by subject areas and the capability of generating a test or exam considering the subject area, level of difficulty, time for answering, assigning weight to one individual question or group of questions, as well as assigning weight to the factor time for answering.

The first step starts when several requests for evolving the framework are sent by the business applications development teams to the framework development team. These requests have descriptions of customer needs for business applications, along with the solution given in terms of architecture and implementation. It is also up to the applications development teams to observe the solution as a set of MMFs and estimate cash flows for each of them.

The *Evaluation Pro* features are:

- *Maintain registers of questions and answers*: questions and answers have to be previously registered, so this feature is essential for the others features can achieve what they propose to do.
- *Maintain registers of one individual appraiser and appraisee and group(s) of them*: evaluations have the form of exams, tests or quizzes, which suggests the existence of the role of an appraiser and an appraisee.
- *Evaluations correction*: should provide immediate feedback for the objective multiple-choice questions to the appraisees and support correction of discursive subjective questions to the appraisers.
- *Questionnaires creation*: creates a list of questions and answers, which will compose an evaluation.
- *Evaluations creation*: association of a questionnaire to one or more appraisers/appraisees.

- *Reports of evaluations statistics*: generate statistics about specific evaluations, incidence of correct answers, average grade of a given appraisee, frequency of use of a given question in evaluations are examples of possible reports.
- *Communication through e-mail to appraisers and appraisees*.

The *Evaluation Pro* entity types are:

- *Appraiser*: User(s) or group(s) of users responsible for creating the evaluation.
- *Appraisee*: User(s) or group(s) of users responsible for answering the evaluation.
- *Evaluation*: Relationship between a questionnaire, an appraiser (who creates the evaluation) and an appraisee (who answers the evaluation).
- *Questionnaire*: Collection of questions and answers.
- *Question*: It belongs to one or more subjects areas and is classified into three types: (1) objective multiple-choice question with only one correct answer, (2) objective multiple-choice question with more than one correct answer and (3) question with discursive and subjective answer. Each question is related to a set of answers.
- *Answer*: It is related to a single question.

Customers of different business applications that are customizations of *Evaluation Pro*, now want to group questions and answers by subject area (**req1**), to assign difficulty levels for each question (**req2**), to assign weights to questions in a given evaluation, for the reason to represent a greater or lesser contribution of a question to the final score (**req3**), and to consider the time for resolution of each question and of the whole evaluation, in computing the score for each individual question and for the evaluation as a whole (questions answered in a shorter time get a higher score than those answered in a longer time) (**req4**).

Each requirement does not necessarily come from a single customer, for example, customers of application A need the new functionalities **req1** and **req4** and customers of application B need the **req2**, **req3** and **req4** ones.

The development team that supports business application A (development team A) understood that two modules will be required to fulfill the new requirements: one module is an improvement on a previously existing software component (responsible for the registration of questions and answers), and the other will be a new module. The former will provide the new attribute "subject area" as a way of grouping related questions, while the latter will provide the inclusion of time as a factor in the computation of the score attribute.

The development team that supports business application B (development team B) decided to implement three modules as the best way to meet the requirements: two of them are improvements in already existing components, namely, the component responsible for registering questions and answers and other one for creating an evaluation, and the third module will be built from the scratch. The first module will include the new "difficulty level" attribute that can be assigned to a question. The second module corresponds to the assignment of weights to questions. The third module is the inclusion of the time to solve a question in computing the score of each question and of the entire evaluation (time as an influencing factor for computing the score).

By following a financial discipline aiming at generating value for the business as soon as possible, development teams A and B designed the modules described above as MMF, whose costs were estimated by the developers as well as the future revenues. Table II shows the cash flow for each MMF.

Henceforth, for the reason of simplicity, the modules implemented by development team A, namely, MMF-A₁ and MMF-A₂, correspond to the new functionalities **req1** and **req4** respectively. For the

development team B, the implemented modules MMF-B₁, MMF-B₂ and MMF-B₃ match respectively to the new functionalities **req2**, **req3** and **req4**.

Table 2. Cash Flow for Each MMF along 3 Years

Period ^a	Business Applications				
	MMF-A ₁	MMF-A ₂	MMF-B ₁	MMF-B ₂	MMF-B ₃
1	-10.0 ^b	-20.0	-5.0	-15.0	-15.0
2	2.0	5.0	0.5	5.0	7.0
3	2.5	6.0	0.6	5.2	7.0
4	3.0	6.5	0.8	5.8	7.0
5	3.5	7.0	1.0	6.0	7.0
6	4.0	8.0	1.3	6.0	7.0
7	4.5	8.5	1.5	6.0	7.0
8	5.0	9.5	2.0	6.0	7.0
9	5.5	10.0	2.2	6.0	7.0
10	5.6	10.0	2.3	6.0	7.0
11	5.7	10.0	2.4	6.0	7.0
12	5.7	10.0	2.5	6.0	7.0

a. A three-month period

b. Monetary values are in thousands of US\$

Based upon the functionalities specified by teams A and B, the new framework features (*Evaluation Pro* new features to be built) are:

- *Maintain registers of subject areas*: Required for functionality **req1**, it will be named as AE1.
- *Chronometer for counting of resolution time of questions and evaluations*: Its construction will enable the functionality **req4** and it will be known as AE3.

Based upon the functionalities elicited by development teams A and B, the existing features to be modified in the framework (*Evaluation Pro* existing features to be evolved) are:

- *Maintain registers of questions and answers*: Because subject area and difficulty level will be attributes of the entity question, adjustments in this feature are required. Such changes are represented as AE2.
- *Evaluations creation*: This feature must be evolved because weights can be assigned to a question of a given evaluation as well as questions can be related to a chronometer with time limit for answering it and a math formula to apply discounts to the maximum score allowed to the question, depending on the time spent to answer it (the more time spending the bigger the discount). The same question could be associated to different weights depending on the evaluation. Likewise, in distinct evaluations, the same question could have different time limits for resolution and different math formulas for score discounting. All of these modifications will be referred as AE4.
- *Evaluations correction*: Modifications, that will be called AE5, due to influence of the time limit for resolution and weight both in the score of an only question and of the whole evaluation.

Table 3. Cash Flow for Each AE along 3 Years

Period ^a	Framework				
	AE1	AE2	AE3	AE4	AE5
1	-20.0 ^b	-30.0	-5.0	-10.0	-9.0
2	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0
5..12	0.0	0.0	0.0	0.0	0.0

a. A three-month period

b. Monetary values are in thousands of US\$

Table 3 shows the cash flow for each AE. In the example, all of them has the duration of only one three-month period to be build and this is the reason why the costs are 0.0 for all the periods than the first.

At the third step of the method, precedence relationships are defined between AE and MMF (AE always comes firstly), AE and AE and between MMF and MMF. Fig. 1 summarizes these dependencies.

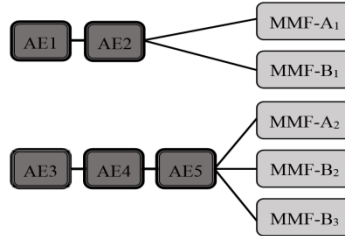


Fig. 1. Precedence relationships between AEs and MMFs.

Finally, at the fourth step, the best sequence of delivery for the framework releases is determined.

Tables 4 and 5 present the SANPV for each MMF and AE. A discount rate of 2 percent per period is used.

Table 4. SANPV for Each MMF along 3 Years

Period ^a	Business Applications				
	MMF-A ₁	MMF-A ₂	MMF-B ₁	MMF-B ₂	MMF-B ₃
1	30.4 ^b	58.3	9.6	41.0	52.5
2	25.4	49.4	7.5	35.5	46.0
3	20.5	40.7	5.5	30.2	39.7
4	15.8	32.2	3.6	25.0	33.5
5	11.2	23.8	1.8	19.8	27.4
6	7.1	16.0	0.2	14.8	21.5
7	3.5	9.1	-0.9	9.9	15.7
8	0.4	2.8	-1.9	5.0	9.9
9	-2.4	-2.7	-2.7	0.3	4.3
10	-4.6	-7.7	-3.2	-4.2	-1.2
11	-6.5	-12.1	-3.6	-8.1	-6.5
12	-7.9	-15.8	-3.9	-11.8	-11.8

a. A three-month period

b. Monetary values are in thousands of US\$

Table 5. SANPV for Each AE along 3 Years

Period ^a	Framework				
	AE1	AE2	AE3	AE4	AE5
1	-19.6 ^b	-29.4	-4.9	-9.8	-8.8
2	-19.2	-28.8	-4.8	-9.6	-8.7
3	-18.8	-28.3	-4.7	-9.4	-8.5
4	-18.5	-27.7	-4.6	-9.2	-8.3
5	-18.1	-27.2	-4.5	-9.1	-8.2
6	-17.8	-26.6	-4.4	-8.9	-8.0
7	-17.4	-26.1	-4.4	-8.7	-7.8
8	-17.1	-25.6	-4.3	-8.5	-7.7
9	-16.7	-25.1	-4.2	-8.4	-7.5
10	-16.4	-24.6	-4.1	-8.2	-7.4
11	-16.1	-24.1	-4.0	-8.0	-7.2
12	-15.8	-23.7	-3.9	-7.9	-7.1

a. A three-month period

b. Monetary values are in thousands of US\$

A practical way to calculate the SANPV is shown in [6].

The SANPV for a module in a given period p is the sum of each cash (value per period, as shown in the lines of the Tables II and III) with a discount rate to bring the cash to a present value, from period 1 until period n , where n is the number of periods between the period 12 (when the lifecycle ends and the software becomes obsolete) and period p (when the module is released), including both periods. As an example, if p is 8 then n would be 5, because there are three periods between periods 12 and 8.

To demonstrate how it works, one can take the data of AE2 from Table 3 to compute the SANPV in period 8.

The cash value for AE2 in period 1 is -30.0, so we have to bring this value to period 8. To do this, we should apply the discount rate above-mentioned of 2%/period. As a result, in period 8, the contribution of period 1 to SANPV amount is -25.6 (from $-30.0/(1,02)^8$). Additionally, we need the contribution of the subsequent periods after period 1: if the module is ready to use in period 8 and it will become obsolete in period 12, we also should consider the contribution of the estimated values in periods 2, 3, 4 and 5 for the SANPV:

- For the value of period 2, NPV is 0.0 (from $0.0/(1,02)^9$).
- For the value of period 3, NPV is 0.0 (from $0.0/(1,02)^{10}$).
- For the value of period 4, NPV is 0.0 (from $0.0/(1,02)^{11}$).
- For the value of period 5, NPV is 0.0 (from $0.0/(1,02)^{12}$).

Thus, the SANPV for AE2 in period 8 is $-26.5 + 0.0 + 0.0 + 0.0 + 0.0$ resulting in -26.5.

Similarly, one can take the data of MMF-A₂ from Table II to compute the SANPV in period 4. Thus, one should take into account the periods 1 to 9 (seven periods between 12 and 4 plus two periods) and the discount rate (2%/period). The contributions of each period are:

- For the value of period 1, NPV is -18.5 (from $-20.0/(1,02)^4$).
- For the value of period 2, NPV is 4.5 (from $5.0/(1,02)^5$).
- For the value of period 3, NPV is 5.3 (from $6.0/(1,02)^6$).
- For the value of period 4, NPV is 5.7 (from $6.5/(1,02)^7$).
- For the value of period 5, NPV is 6.0 (from $7.0/(1,02)^8$).
- For the value of period 6, NPV is 6.7 (from $8.0/(1,02)^9$).
- For the value of period 7, NPV is 7.0 (from $8.5/(1,02)^{10}$).
- For the value of period 8, NPV is 7.6 (from $9.5/(1,02)^{11}$).
- For the value of period 9, NPV is 7.9 (from $10.0/(1,02)^{12}$).

Therefore, the SANPV for MMF-A₂ in period 4 is 32.2, the sum of all values listed above.

For simplicity, the following conditions were assumed for the project:

- The effort to develop a single AE or MMF lasts exactly only one period of three months.
- The framework development team implements one AE at a time, even when there is no dependency relationship between AE's. In other words, the team cannot be divided up in such a way that more than one AE might be developed at same time (in parallel).
- Likewise, development teams A and B implement one MMF at a time. That is, the business application A modules MMF-A₁ and MMF-A₂ cannot be implemented at the same time, although they are independent (there's no precedence relationship between them). This is also true for MMF-B₁, MMF-B₂ and MMF-B₃.
- However, as the A, B and framework development teams work independently from each other and do not share resources, it is perfectly possible that the development efforts of AE3 (also both AE4 and AE5), MMF-A₁ and MMF-B₁ can occur in parallel.

Based on Fig. 1, Table 6 shows the SANPV (leftmost column) for all possible delivery sequences.

Table 6. SANPV for Each Sequence

Period ^a							SANPV
1	2	3	4	5	6	7	
AE1	AE2	AE3 MMF-A ₁ MMF-B ₁	AE4	AE5	MMF-A ₂ MMF-B ₂	MMF-B ₃	2.0 ^b
AE1	AE2	AE3 MMF-A ₁ MMF-B ₁	AE4	AE5	MMF-A ₂ MMF-B ₃	MMF-B ₂	2.9
AE1	AE3	AE2	AE4 MMF-A ₁ MMF-B ₁	AE5	MMF-A ₂ MMF-B ₂	MMF-B ₃	-4.2
AE1	AE3	AE2	AE4 MMF-A ₁ MMF-B ₁	AE5	MMF-A ₂ MMF-B ₃	MMF-B ₂	-3.3
AE1	AE3	AE4	AE2	AE5 MMF-A ₁ MMF-B ₁	MMF-A ₂ MMF-B ₂	MMF-B ₃	-10.2
AE1	AE3	AE4	AE2	AE5 MMF-A ₁ MMF-B ₁	MMF-A ₂ MMF-B ₃	MMF-B ₂	-9.3
AE1	AE3	AE4	AE5	AE2 MMF-A ₂ MMF-B ₂	MMF-A ₁ MMF-B ₁	MMF-B ₃	-2.7
AE1	AE3	AE4	AE5	AE2 MMF-A ₂ MMF-B ₂	MMF-A ₁ MMF-B ₃	MMF-B ₁	2.0
AE1	AE3	AE4	AE5	AE2 MMF-A ₂ MMF-B ₃	MMF-A ₁ MMF-B ₁	MMF-B ₂	-0.9
AE1	AE3	AE4	AE5	AE2 MMF-A ₂ MMF-B ₃	MMF-A ₁ MMF-B ₂	MMF-B ₁	-9.7
AE3	AE1	AE2	AE4 MMF-A ₁ MMF-B ₁	AE5	MMF-A ₂ MMF-B ₂	MMF-B ₃	-3.9
AE3	AE1	AE2	AE4 MMF-A ₁ MMF-B ₁	AE5	MMF-A ₂ MMF-B ₃	MMF-B ₂	-3.0
AE3	AE1	AE4	AE2	AE5 MMF-A ₁ MMF-B ₁	MMF-A ₂ MMF-B ₂	MMF-B ₃	-9.9
AE3	AE1	AE4	AE2	AE5 MMF-A ₁ MMF-B ₁	MMF-A ₂ MMF-B ₃	MMF-B ₂	-9.0
AE3	AE1	AE4	AE5	AE2 MMF-A ₂ MMF-B ₂	MMF-A ₁ MMF-B ₁	MMF-B ₃	-2.4
AE3	AE1	AE4	AE5	AE2 MMF-A ₂ MMF-B ₂	MMF-A ₁ MMF-B ₃	MMF-B ₁	2.3
AE3	AE1	AE4	AE5	AE2 MMF-A ₂ MMF-B ₃	MMF-A ₁ MMF-B ₁	MMF-B ₂	-0.6
AE3	AE1	AE4	AE5	AE2 MMF-A ₂ MMF-B ₃	MMF-A ₁ MMF-B ₂	MMF-B ₁	4.4
AE3	AE4	AE1	AE2	AE5 MMF-A ₁ MMF-B ₁	MMF-A ₂ MMF-B ₂	MMF-B ₃	-9.7

AE3	AE4	AE1	AE2	AE5 MMF-A ₁ MMF-B ₁	MMF-A ₂ MMF-B ₃	MMF-B ₂	-8.8
AE3	AE4	AE1	AE5	AE2 MMF-A ₂ MMF-B ₂	MMF-A ₁ MMF-B ₁	MMF-B ₃	-2.2
AE3	AE4	AE1	AE5	AE2 MMF-A ₂ MMF-B ₂	MMF-A ₁ MMF-B ₃	MMF-B ₁	2.5
AE3	AE4	AE1	AE5	AE2 MMF-A ₂ MMF-B ₃	MMF-A ₁ MMF-B ₁	MMF-B ₂	-0.4
AE3	AE4	AE1	AE5	AE2 MMF-A ₂ MMF-B ₃	MMF-A ₁ MMF-B ₂	MMF-B ₁	4.6
AE3	AE4	AE5	AE1 MMF-A ₂ MMF-B ₂	AE2 MMF-B ₃	MMF-A ₁ MMF-B ₁		33.7
AE3	AE4	AE5	AE1 MMF-A ₂ MMF-B ₃	AE2 MMF-B ₂	MMF-A ₁ MMF-B ₁		24.1

a. A three-month period

b. Monetary values are in thousands of US\$

It is very important to realize that does not matter if the project did not recover the investment through generation of profits (even financial loss), by observing a negative SANPV along a three-year period. That is because future customizations of the framework are possible, and this fact would increase the amount of MMF, so generating increased revenue (AE is a cost element only).

What really should be observed is the comparison between the SANPV of each sequence, so that the sequence with the SANPV greatest value should have priority to be implemented.

As noticed in the table VI, the SANPV greatest value is 33.7 that means it is clear that the features that will provide greater return on investment are AE2, AE3 and AE5 and these will compose the next release of the framework.

5. Conclusion

This paper presented a method that is an adaptation of the original IFM as proposed by Denne and Cleland-Huang [7]. The original IFM considers that at any given time only a single software unit (AE or MMF) is being developed. The adapted IFM allows for the existence of one framework development team and several business application development teams. Moreover, each team may be involved with the development of one software unit. As these teams work independently, many software units may be developed concurrently. This is constrained by their precedence relations.

The method allows an integrated view of both framework and business application development processes. Moreover, it establishes the implementation order for both framework and business applications, which maximizes the global financial return yielded by the software development effort.

The method is easy applied to real world situations, as it requires the use of concept and techniques that are usually mastered by project managers. It is especially useful to support the management framework factories that are been built in large companies in specialized fields such as oil and gas, utilities and e-commerce.

All of this is made clear with the help of a reasonably complex example based on projects run by the software factories of an Oil&Gas company.

Acknowledgment

The authors wish to thank Petr leo Brasileiro S.A. (Petrobras) and its Provider Center of Software Solutions at ICT department, especially the development team for *Avalia* software, a domain framework to help developers to build their own applications based on questionnaires, quizzes and surveys. *Avalia's* Product Owner (PO), Sandro Raphael de Oliveira Paiva, with his generosity, kindness and readiness, explained us the business process and system architecture supported by such a software.

References

- [1] Edwin, N. M. (2014) Software frameworks, architectural and design patterns. *Journal of Software Engineering and Applications*. Retrieved 2014, from <http://dx.doi.org/10.4236/jsea>
- [2] Adair, D. (1995). Building object-oriented frameworks, *AIExpert*, Feb. 1995.
- [3] Johnson, R., & Foote, B. (1988). Designing reusable classes, *J. Object-Oriented Program*, 2(1), 22–35, 1988.
- [4] Sparks, S., Benner, K., & Faris, C., Managing object-oriented framework reuse. *IEEE Comp.*, 29(9), 52–62, 1996.
- [5] Alencar, A. J., Doria Jr, J. V., Schmitz, E. A., & Correa, A. L. (2012). On the merits and pitfalls of the incremental funding method and its software project scheduling algorithms. *Communications in Computer and Information Science*, 292, 493–502.
- [6] Denne, M., & Cleland-Huang, J. (2004). The incremental funding method: Data-driven software development. *IEEE Software*, 21, 39–47.
- [7] Denne, M., & Cleland-Huang, J. (2003). *Software by Numbers: Low-Risk, High-Return Development*. Prentice Hall.



Andre de Souza Andrade was born in 1975. He holds a B.Sc. in computer science from Federal University of Pernambuco (UFPE) and a Lato Sensu degree in information systems with focus on internet systems from Federal University of Rio de Janeiro (UFRJ).

He has been a practitioner in software engineering for 12 years. Along 8 years he has worked in SAP ERP projects and related technologies. Recently, his studies are focused on software release planning that uses both formal and human intuitive approaches.

Eber Assis Schmitz is a professor of computer science with the Federal University of Rio de Janeiro (UFRJ). He holds a B.Sc. in electrical engineering from the Federal University of Rio Grande do Sul (FURG), an M.Sc. in electrical engineering from UFRJ and a Ph.D. in computer science from the Imperial College of Science, Technology and Medicine, London, England. His research interests include software development modeling tools, business process modeling and stochastic modeling.

Antonio Juarez Alencar is a researcher with the Federal University of Rio de Janeiro (UFRJ), Brazil. He received his B.Sc. in mathematics and M.Sc. in system engineering and computer science from UFRJ. He holds a D.Phil. in computer science from Oxford University, England. His research interests include economics of software engineering, IT strategy and risk analysis.

Alexandre Luis Correa is a professor of computer science with the State of Rio de Janeiro Federal University (UNIRIO). He holds a B.Sc. in mathematics and an M.Sc. and a D.Sc. in system engineering and computer science from the Federal University of Rio de Janeiro (UFRJ). His research interests include reverse engineering, system validation and software development modeling tools.