Towards a Maturity Model for a Reuse Incremental Adoption

Vinicius Cardoso Garcia[1], **Daniel Lucrédio**[2], **Alexandre Alvaro**[1], **Eduardo Santana de Almeida**[1], **Renata Pontin de Mattos Fortes**[2], **Silvio Romero de Lemos Meira**[1]

[1] Informatics Center – Federal University of Pernambuco
C.E.S.A.R – Recife Center for Advanced Studies and Systems
Recife, Pernambuco

{vcg,aa2,esa2,srlm}@cin.ufpe.br

[2]Institute of Mathematical and Computing Sciences
São Paulo University, São Paulo, Brazil

{lucredio,renata}@icmc.usp.br

Abstract. *Software has been reused in applications development ever since programming started. However, the reuse practices have mostly been ad hoc, and the potential benefits of reuse have never been fully realized. Systematic reuse offers the greatest potential for significant gains in software development productivity and quality. Organizations are looking for ways to develop a software reuse program. The strategy for adopting a reuse technology should be based on a vision for improving the organization's way of doing business. Thus, this paper presents a Reuse Maturity Model proposal, describing consistence features for the incremental reuse adoption.*

1. Introduction

As has been frequently discussed [Biggerstaff and Richter 1987, Frakes and Isoda 1994, Lim 1994, Rine and Nada 2000b, Poulin 2006], the practice of software development has become increasingly worried by high cost, poor productivity, and poor or inconsistent quality. One of the reasons for this problem is the insistence on the part of many software development organizations to develop from the ground up similar systems over and over again, rather than to treat their previous experiences and previously-developed systems as assets to be captured, created, and evolved so that they can contribute directly to future development activities [Rine and Nada 2000b].

Software reuse, the use of existing software artifacts or knowledge to create new software, is a key method for significantly improving software quality and productivity [Frakes and Isoda 1994]. Thus, reuse has been advocated as a means of revolutionizing the development process.

Although reusability is a big challenge on software development area, its promise has been largely unfulfilled. The main inhibiting factors have been the absence of a clear reusability strategy and the lack of specific top-management support, which can lead to resistance from project managers and programmers [Biggerstaff and Richter 1987, Frakes and Fox 1995, Moore 2001, Morisio et al. 2002, Rine 1997a].

A reuse adoption model helps an organization to understand how reuse will change the way it does business, and how it should plan for that change [Wartik and Davis 1999].

The recent interest in characterizing reuse with maturity models and adoption processes is a clear sign of progress toward making reuse a natural part of development. Reuse maturity models provide the necessary framework for the development of tools and methods to aid in the reuse adoption or to support in the organization development process [Griss 1994, Prieto-Díaz 1993, Rine and Nada 2000a].

In this context, this paper describes the initial RiSE[1] Maturity Model, which provides a structured set of reuse engineering, management and support practices related to the implementation of a software reuse program in an organization.

This paper gives an overview of the RiSE Maturity Model, which has been developed within the RiSE project [Almeida et al. 2004]. More precisely, this paper describes: **(i)** other reuse-related maturity models, which have been studied as background information; **(ii)** the approach taken to develop the Maturity Model; and **(iii)** the reuse maturity levels inside the Maturity Model.

The remainder of the paper is organized as follows: Section 2 present the problem related with reuse adoption by software development organizations. Section 3 discusses related work. The proposed approach is described in details in Section 4. Finally, the concluding remarks and future work are presented in Section 5.

2. The problem

Many software development organizations believe that investing in software reuse will improve their process productivity and product quality, and are planning or developing a reuse program [Frakes and Isoda 1994, Tracz 1990]. Unfortunately, there is still not enough data available on the state-of-the practice of utilizing or managing software reuse. The majority of current information available on software reuse models comes from the literature [Card and Comer 1994, Frakes and Fox 1995, Rine and Sonnemann 1998, Rine and Nada 2000b]. After 3 years of experience in industrial reuse projects, the RiSE initiative identified that a critical problem in today's practice of reuse is a failure to develop necessary details to support valid software reuse models. The existing models, including many important and well established works, as described in section 3, do not present such details, which makes their practical usage difficult.

Another problem is related to the question: *what kind of assets can be reusable?* Some researchs [Frakes and Fox 1995, Morisio et al. 2002, Rine 1997a] show that software reuse can happen in others phases of the development cycle, such as analysis and project, obtaining more benefits than source code level. Moreover, high level abstractions can be useful for analysts and designers allowing that previous experiences and knowledge can be reused reducing the probability of risks.

Our research, reported in this paper, investigated the success factors for software reuse, the way they impact in a software reuse program, and how they can be used to construct a framework for a reuse maturity model. This work is based on empirical data collected in a survey that investigated the reuse situation in several organizations in Brazil [Lucrédio et al. 2007], which covered many factors related to software reuse.

[1]Reuse in Software Engineering Group, `http://www.rise.com.br`

3. Reuse Adoption Models: A Brief Survey

Some software reuse models have been proposed to face the reuse adoption problem. The vast majority of reuse models attempt to provide some measure of a reuse program success in a systematic way. In this section, we present a brief survey on these approaches.

Holibaugh et al. [Holibaugh et al. 1989] presented the first cost model for software reuse developed at *Software Engineering Institute* (SEI). In this work, Holibaugh et al. described a framework to determine a cost-effective approach to start the reuse adoption. For this framework, it can be noticed: a clear evolution; fulfillment of existent gaps; and a natural ripening in the reuse adoption models. Phases and activities, such as analysis and domain engineering, as well as a well defined software development methodology are considered as a primordial condition for a systematic reuse process in an organization.

In 1991, Koltun and Hudson [Koltun and Hudson 1991] presented the first version of the Reuse Maturity Model (RMM). The model was specified through workshops accomplished by the *Software Productivity Consortium* (SPC). The model, in fact, provides a concise form of obtaining information on reuse practices in organizations. The model is composed of five levels and ten dimensions or aspects of reuse maturity were also enumerated. The main obstacles to reach the highest reuse levels, as pointed out by Koltun and Hudson, were: Cultural, Institutional, Financial, Technical and Legal. This model was not applied in real case studies but will be considered as the main insight for the Margaret Davis work [Davis 1992].

Starting from his experience and the best features of four success cases, Prieto-Díaz defined an incremental model for the implementation of software reuse programs [Prieto-Díaz 1991]. The approach was practical, effective and with potential to turn reuse into a regular practice in the software development process. Prieto-Díaz also points out the support of the high administrative management as one of the main factors for the success in reuse programs, because those programs demanded changes in the way software is developed.

One of the most interesting features in Prieto-Díaz's model is the definition of an essential organizational structure, in the first stage of the model implementation, containing teams for: assets management; assets identification and qualification; assets maintenance; assets development; support to the reuse program through consultancy and training; and classification of the assets in the repository system. These teams perform the basic roles in the reuse program.

In November 1992, the *Fifth Workshop on Institutionalizing Software Reuse* (WISR) was held. Margaret Davis presented the reuse maturity model of the STARS project [Davis 1992]. The first reuse maturity model, presented by Koltun and Hudson had important influence in this one, because Hudson participated directly in the STARS project. This maturity model is the base for organizations to formulate their short and long term strategies to improve the level of reuse practice in their business domains. Moreover, Margaret Davis believes that the maturity model can be used with other principles, such as a reuse practice level evaluation tool, or a way to encourage the reuse adoption through incentive programs.

The mains issues in Margaret Davis model is the high up-front risk of reuse adoption, because a significant initial investment is needed. However, one positive point is that

the model was designed to be independent of a particular development model.

In the next year, Ted Davis [Davis 1993] presented the *Reuse Capability Model* (RCM), an evolution of the STARS' reuse maturity model. RCM aids in the evaluation and planning for improvements in the organization's reuse capability. RCM is used together with the reuse adoption process defined by SPC [SPC 1993]. The reuse adoption process is a solution to implement a reuse program and it is based on the implementation model defined by Prieto-Díaz [Prieto-Díaz 1991].

The RCM has two components: an assessment model and an implementation model. The **assessment model** consists of a set of critical success factors to assess the present state of its reuse practice. From this assessment, the organization will get a list of its strengths and a list of potential improvement opportunities. The **implementation model** helps in prioritizing the critical success factor goals by partitioning them into a set of stages. This model will be considered as the main insight for the Wartik and Davis [Wartik and Davis 1999] work.

The utilization of a specific reuse adoption process is a great issue of Ted Davis work. RCM, in conjunction with this adoption process, helps the organization in business decisions and in how its practices work in its development process. But, according to Rine and Sonnemann, in their study of software reuse investment success factors [Rine and Sonnemann 1998], RCM proved to be unstable. It is a hierarchical model with each level building on the previous level. It produced substantially different results depending on whether or not the level questions were answered bottom up or top down. However, the six measurements used to evaluate software reuse capability did not correlate very well.

Not directly related of reuse adoption model we can notice the standard ISO/IEC 12207, addressed to aiming the organizations in their software development process. The standard ISO/IEC 12207 - Software Life-Cycle Process [ISO/IEC 1998] offers a framework for software life-cycle processes from concept through retirement. It is especially suitable for acquisitions because it recognizes the distinct roles of acquirer and supplier. ISO/IEC 12207 provides a structure of processes using mutually accepted terminology, rather than dictating a particular life-cycle model or software development method. Since it is a relatively high-level document, 12207 does not specify the details of how to perform the activities and tasks comprising the processes.

Wartik and Davis [Wartik and Davis 1999] present a new version of the reuse adoption model of SPC [Davis 1993]. The model is based on a set of phases that help the organization to measure its progress towards the reuse adoption. Each phase has specific goals, integrated into the Synthesis methodology [Burkhard 1993]. Besides, the phases are spent in such a form that the organization can avoid the risks, or at least, to reduce them significantly in the reuse adoption program through the selection of the main features of each phase to achieve their goals.

The main motivations for a new model, according to the lessons learned after the observation of the organizations in several stages in the reuse adoption process were: **reduce the initial risk in the reuse adoption**, so that organizations can recognize the need to define the reuse goals, making it easier to commit with a reuse adoption model that demands a commitment of resources incrementally with base in a constant evolution and

understanding of their benefits; and, **integration with a reuse-based software development process**, merging the reuse adoption model with the Synthesis methodology of SPC, reducing the number of decisions related to reuse that the organization should take, making the efforts for the adoption simpler.

Another reference model for software reuse called Reuse Reference Model (RRM) was presented by Rine and Nada [Rine and Nada 2000a]. RRM incorporates both technical and organizational elements that can be applied to establish a successful practice of software reuse in the organization. The technical elements consist of technologies that support reuse, such as *CASE* tools and a software reuse process, among others. Organizational elements include management of the reuse program, market analysis, financing and training.

In order to increase their software development capabilities, Brazilian software industries and research universities are working cooperatively to implement a strategy aiming to improve software processes of Small and Medium-size Enterprises Brazilian organizations since 2003. The main goal of this initiative is to develop and disseminate a Brazilian software process model (named MPS Model) [SOFTEX 2007] based on software engineering best practices and aligned to Brazilian software industry realities. The focus of the MPS Model is on small settings, since it provides mechanisms to facilitate software process improvement (SPI) implementation of the most critical software processes. The adequate implementation of such processes promotes subsequent SPI implementation cycles and software process maturity growth.

However, such as the ISO/IEC 12207, MPS does not have some practices related to reuse activities or to aiming to adopt reuse practices in the software development process. Therefore, we believe that can be possible for some reuse practices to be integrated to these models, specially to ISO/IEC 12207 (the basis of MPS and CMMI [Chrissis et al. 2004]) to specify a new strategy to help in the improvement of productivity and quality in the software development process in organizations.

Based on the research results and case studies, Rine and Nada conclude that the level of reuse, as defined in RRM, determines the capability of improvements in the productivity, quality and time-to-market of the organization.

3.1. Discussion

From this brief survey we may notice that it is clear that development *"for"* (domain engineering) and *"with"* (application engineering) reuse are needed. Another factor is the concern with reuse since early stages of the software life cycle, beginning with the elaboration of the business plan. Reuse should be considered in the first phases of the software development cycle.

The reuse community agrees [Davis 1993, Rine and Nada 2000a, SPC 1993, Wartik and Davis 1999] that characterizing reuse with maturity models and adoption processes is a clear sign of progress toward making reuse a natural part of development. As shown in this section, there are some experiences and projects involving reuse adoption models and programs. However, the organizations are still unsure of adopting a reuse program because there is not a widely accepted model. Models proposed until now fail in transferring the reuse technology to the entire organization, in an incremental and systematic way. The main problem is that most works classify the reuse adoption as an atomic

initiative, and introducing atomic, radical changes is not appealing to most organizations. A more gradual evolution is more suited to reuse [Rine and Nada 2000b]. Thus, through this survey we identify the main requirements of some reuse adoption models to specify an effective reuse maturity model, in order to implement a systematic and incremental approach to reduce the risks and improve the possibilities of success in this journey. This model will be presented in the next section.

4. Towards a RiSE Maturity Model

The RiSE Maturity Model was developed during the RiSE project [Almeida et al. 2004] through discussions with industry practitioners. Initially, the model included four perspectives addressing organizational, business, technological and process issues. The idea is to provide the organizations the possibilities to develop and improve these perspectives separately. However, our experience in the RiSE project[2] and work reports in the literature showed that the organizational, business and technological perspectives are interrelated, and have to be implemented in parallel, while the processes perspectives are relevant for highly mature organizations with respect to software reuse program adoption. Therefore, the individual perspective are combined in the final version of the maturity model and the activities related to their implementation are defined in terms of specific practices.

4.1. Model Structure

The main purpose of the RiSE Maturity Model is to support organizations in improving their software development processes. In particular, the model has to serve as a roadmap for software reuse adoption and implementation. It aims at facilitating the improvement of the engineering practices with respect to their effectiveness and efficiency, performing reuse activities properly and achieving the quality goals for the products.

Current models for process improvements like CMMI and ISO 9001:2000 typically address organizational processes. Although they also discuss software engineering activities, they do not provide much insight in the way of performing these tasks, neither discuss how particular technologies could support the improvement initiatives.

The model structure intend to be flexible, modular and adaptable to the needs of the organizations that will use them. Thus, the model was based in two principles: modularity and responsibility. Modularity, in the sense of process with less coupling and maximum cohesion. Responsibility, in the sense of the possibility to establish one or more (team) responsible for each process, or activity (perspectives and factors). This structure easiest the model implementation in places where various professionals can be involved in the reuse adoption.

The RiSE Maturity Model includes: **(i)** Reuse practices grouped by perspectives (Organizational, Business, Technological and Processes) [Brito et al. 2006, Lucrédio et al. 2007] and in organized levels representing different degrees of software reuse achieved; and, **(ii)** Reuse elements describing fundamental parts of reuse technology, e.g. assets, documentation, tools and environments.

The maturity levels provide general characterization of the organizations with respect to the degree of reuse adoption and implementation, i.e. a maturity level indicates what reuse practices and assets are expected to be in place in an organization.

[2]The RiSE group has been involved in 4 industrial projects related to reuse adoption since 2004.

The practices implemented at the lower maturity levels are a basis for the implementation of the activities at the upper levels. This means that an organization at a certain level of maturity has to successfully implement all the practices from this level and the ones below it. The same is valid for the reuse elements associated with a level.

The following maturity levels are defined: **Level 1**: Ad-hoc Reuse; **Level 2**: Basic Reuse; **Level 3**: Initial Reuse; **Level 4**: Integrated Reuse; and, **Level 5**: Systematic Reuse. The levels are defined in details in section 4.2. Each level consists of a list of reuse practices and reuse elements associated to them.

Goals are defined for each maturity level. They are used to guide the assessment of the implementation of the maturity model. More precisely, while the execution of the practices defined at a maturity level could vary, the achievement of the goals for that level and the levels bellow is mandatory to conclude that the level is achieved.

4.2. RiSE Maturity Model Level Definitions

The RiSE Maturity Model consists of the following elements: Maturity Levels, Goals assigned to each level, Perspectives (Organizational, Business, Technological and Processes) and Practices grouped in levels and perspectives.

Five levels are identified in the RiSE Maturity Model as shown on Figure 1.

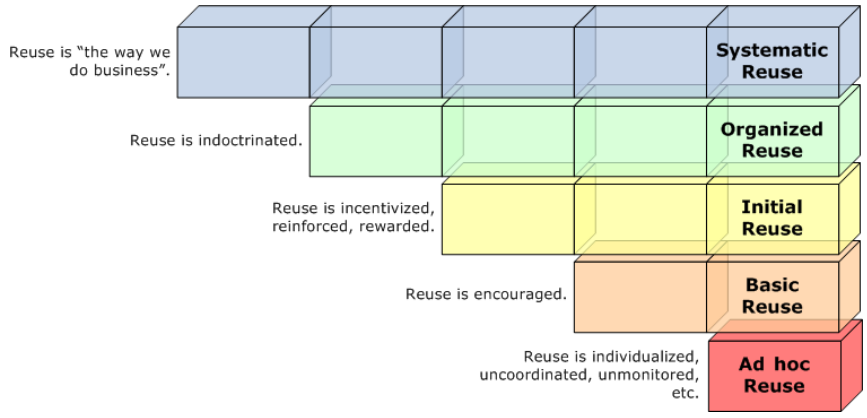

Figure 1. RiSE Maturity Model levels

A maturity level represents the level of reuse adoption. Each level has particular goals associated to it, which are used to determine whether it is completely achieved. The maturity levels also allow foreseeing which reuse-related activities will be performed in an organization in the future. However, an organization does not need to evolve one stage at a time. It may decide its own evolution path, according to strategical and/or business goals, by choosing which factors should be improved first, until the organization reaches the desired reuse capability. The RiSE Maturity levels are presented next.

Level 1: Ad hoc Reuse.

Traditional software development is performed. Reuse practices are sporadically used or not used at all and is discouraged by management. This practices are performed as an individual initiative (personal goal; as time allows). The costs of reuse are unknown.

Level 2: Basic Reuse.

This level is characterized by a basic usage of potentially reusable assets. It encompasses basic reuse-oriented engineering activities. The assets are used for aiding in the implementation and production of components and documentation. Simple tools and techniques are used to develop reusable assets (documents, design, code, etc). Typically, technical assets are developed, which, in this level of maturity, include all the requirements of the system with no distinction between business and domain specific aspects.

Code and documentation are generated by methods of reuse-based tools or available COTS supporting this operation, but not developed by the organizations. This means that the assets reuse is performed by a tool and no particular engineering effort on defining such asset is needed. Developers modify the generated assets manually to complete the implementation of the application.

From the business perspective, the benefits for an organization applying reuse at this level consist in acquiring experience in software and system design reuse.

The following goals and practices are defined at level 2.

- Goal 1: Using reuse best practices to designing and implementing the software product.
- Goal 2: Use the technical assets to build up software (code and documentation).

Level 3: Initial Reuse.

At this level a separation between business and domain-related assets is introduced. The objective is to maintain the implementation issues independent from the business issues in order to increase the efficiency of the development process by reusing assets for projects of a different nature which have similar business requirements. This is essentially important for system families. Additionally, initial steps towards automating the engineering process are made.

Reuse practices are standardized and deployed in the whole organization (institutionalized). Engineering process knowledge is stored in a reuse repository (asset manager) [Koltun and Hudson 1991]. Metrics on reuse activities are also defined and analyzed to ensure their good performance with respect to predefined organization-wide policies.

The key difference between level 2 and level 3 is that at level 2 any project might implement a different approach to reuse provided that the intent of the specified practices is fulfilled. At level 3 projects are getting some reuse guidelines and assets from the organization and then adapting them to the project, following organization's conventions, using them and providing feedback to the organization in terms of lessons learned and suggestions for improving the practice or asset descriptions.

From the business perspective, the gained benefits for an organization applying software reuse when reaching this level consist in establishing the basis of formalizing and maintaining organizational know-how and application domain knowledge. Another benefit is standardizing the way-of-doing of all the projects, making reuse processes more mature in the sense they are formally described, followed by all projects and quality ensured.

The following goals and practices are defined at level 3.

- Goal 1: Separate business-specific aspects.

- Goal 2: Institutionalize reuse practices and assets.
- Goal 3: Use software reuse project's defined process.

Level 4: Organized Reuse.

The Organized Reuse level is characterized by a better integration of all reuse abstractions levels. At the highest abstraction level, reuse is indoctrinated. Staff members know the reuse vocabulary and have reuse expertise. Reuse occurs across all functional areas.

At the level 4, domain engineering is performed. Reuse-based processes are in place to support and encourage reuse and the organization has focus on developing families of products. The organization has all data needed to decide which assets to build/acquire, because it has a reuse inventory organized along application-specific lines.

From the point of view of the processes, in level 4 the reuse practices inside the projects have started to be quantitatively managed. Metrics are consistently collected and analyzed as defined by the organization.

From the business perspective, all costs associated with an asset's development and all savings from its reuse are reported and shared. Additionally, reuse practices and assets progress and quality in all projects are statistically controlled through standardized metrics that leads to a better control and estimates of the projects objectives. In this level, rework efforts are also reduced due to early detection.

The following goals and practices are defined at level 4.

- Goal 1: Enhance the organization competitive advantage.
- Goal 2: Integrate reuse activities in the whole software development process.
- Goal 3: Ensure efficient reuse performance.

Level 5: Systematic Reuse.

In the Systematic Reuse level, the whole organization's knowledge is planned, organized, stored and maintained in a reuse inventory (asset manager), and used with focus in the software development process. All major obstacles to reuse have been removed. All definitions, guidelines, standards are in place, enterprise-wide.

Domain engineering [Almeida 2007] practices are put in place. In fact, all reusable assets and knowledge are continuously validated in order to make strategic assets reusable. All software products are generalized for future reuse. Domain analysis is performed across all product lines. All system utilities, tools, and accounting mechanisms are instrumented to track reuse.

From the reuse inventory perspective, the development process supports a planned activity to acquire or develop missing pieces in the catalog. From the technological perspective, the organization has automated support integrated with development process.

From business perspective the organization maximizes the benefits of having implemented the whole software reuse approach. The organization is able to express its system development know-how in the form of reusable assets and even more, they are the start point for a rapid and automatic production of the implementation. This reduces effort consumption and accelerates time to market. All costs associated to a product line or a particular asset and all savings from its reuse are reported and shared.

The following goals and practices are defined at level 5.

- Goal 1: Reuse is *"the way we do business"*.
- Goal 2: Establish and maintain complete reuse-centric development.

4.3. Perspectives and Factors

After an extensive literature review [Almeida et al. 2005, Frakes and Fox 1995, Morisio et al. 2002, Rine 1997b, Rine 1997a] and from our experience in reuse projects, we identified some factors related to software reuse [Brito et al. 2006, Lucrédio et al. 2007], that were considered as a basis for this maturity model specification, in order to guide the organizations in the reuse evaluation and/or adoption.

Four perspectives are defined in the RiSE Maturity Model: **Organizational**, **Business**, **Technological** and **Processes**.

The Organizational perspective addresses activities that are directly related to management decisions necessary to setup and manage a reuse project. The business perspective addresses issues related to the business domain and market decisions for the organization. The technological perspective covers development activities in the software reuse engineering discipline and factors related to the infrastructure and technological environment. The processes perspective includes only activities that support the implementation of the engineering and the project management practices.

In the RiSE Maturity Model, fifteen factors were considered, divided into these four perspectives. This division is useful for organizations, which can put special focus on different parts of reuse adoption, one perspective at a time. Another possibility is to assign particular teams for each perspective, according to technical skills and experience, so that each group of factors may be dealt with simultaneously by specialized professionals.

Figure 2 shows the factors related to the organizational perspective and their distribution across the RiSE Maturity Model levels. Figure 3 shows the factors related to the business perspective. Figure 4 shows the factors related to the technological perspective. And finally, Figure 5 shows the factors related to the processes perspective.

Factors of influence	Levels				
	1. Ad-hoc	2. Basic	3. Initial	4. Organized	5. Systematic
Planning for reuse	• Nonexistent.	• Grassroots activity. • Reuse is viewed as single-point opportunities. • Individual achievements are rewarded.	• Targets of opportunity • Organization responsible for reuse. • A key business strategy.	• Business imperative. • Reuse occurs across all functional areas.	• Part of a strategic plan. • Discriminator in business success.
Software reuse education	• Lack of expertise by the staff members (engineers and managers). • Frequent resistance to reuse.	• Basic definitions of reuse are agreed upon.	• The staff has the expertise and how to obtain benefits with reuse.	• The staff members know the reuse vocabulary and have reuse expertise.	• All definitions, guidelines, standards are in place, enterprise-wide.
Legal, Contractual, Accounting considerations	• Inhibitor to getting started.	• Internal accounting scheme for sharing costs, allocating benefits.	• Data rights and compensation issues resolved with customer.	• Royalty scheme for all suppliers and customers.	• Software treated as key capital asset.
Funding, Costs and Financial Features.	• Costs of reuse are unknown.	• Costs of reuse are "feared".	• Payoff of reuse is "known" and understood for a given domain. • Investments made in reuse, payoffs expected. • Costs of reuse are "known".	• All costs associated with an asset's development and all savings from its reuse are reported and shared.	• All costs associated to a product line or a particular asset and all savings from its reuse are reported and shared.
Rewards and incentives	• Reuse is discouraged by management.	• Reuse is encouraged.	• Reuse is motivated, reinforced, rewarded.	• Reuse is indoctrinated.	• Reuse is "the way we do business".
Independent reusable assets development team.	• Individual initiative (personal goal; as time allows).	• Shared initiative.	• Dedicated individual.	• Dedicated group.	• Corporate group (for visibility not control) with division liaisons.

Figure 2. RiSE Maturity Model Levels: Organizational Factors

Factors of influence	Levels				
	1. Ad-hoc	2. Basic	3. Initial	4. Organized	5. Systematic
Product family approach	• Isolated products. • No family product approach.	• Common features and requirements across the products. • Commonalities and reuse possibilities were identified.	• Product line domain analyses performed.	• Focus on developing families of products. • Domain Engineering performed.	• Domain analyses performed across all product lines. • Product family approach.
Software reuse education	• Chaotic development process; unclear where reuse comes in.	• Reuse questions raised at design reviews (after the fact). • Development process defined (some reuse activity indications).	• Design emphasis placed on reuse of off-the-shelf parts. • Product line domain analyses performed. • Shared understanding of all the activities needed to support reuse.	• Focus on developing families of products. • Reuse-based processes are in place to support and encourage reuse. • Domain Engineering performed.	• All software products generalized for future reuse. • Domain analyses performed across all product lines. • Product family approach.

Figure 3. RiSE Maturity Model Levels: Business Factors

Factors of influence	Levels				
	1. Ad-hoc	2. Basic	3. Initial	4. Organized	5. Systematic
Repository systems usage	• Salvage yard (No apparent structure to collection).	• Catalog identifies language- and platform-specific parts. • Simple structures like Concurrent Versions Systems. • Considered mainly source code.	• Catalog includes generic data processing functions. • Considered software components, reports and document models.	• Catalog organized along application-specific lines. • Have all data needed to decide which assets to build/acquire. • Considered screen generators, database elements and test cases.	• Planned activity to acquire or develop missing pieces in catalog. • Considered all artifacts of software development life cycle.
Technology support	• Personal tools, if any.	• A collection of tools, e.g. CM, but not specialized to reuse. • General-purpose analyzers combined to assess reuse levels.	• Classification aids & synthesis aids. • Standardization on components and architecture. • Tools customized to support reuse.	• Digital library separate from development environment.	• Automated support integrated with development system. • Fully integrated with development and reporting systems.

Figure 4. RiSE Maturity Model Levels: Technological Factors

All the perspectives describe activities specific to software reuse. This means that activities which are typical for traditional software development are not included in this model.

The RiSE Maturity Model: **(i)** Supports constant evolution, in an incremental way, with five levels of reuse maturity; **(ii)** Defines reuse-specific engineering, management and support practices that are expected to be put in place at each level of maturity so that any organization can adopt the RiSE Maturity Model.

Factors of influence	Levels				
	1. Ad-hoc	2. Basic	3. Initial	4. Organized	5. Systematic
Quality models usage	• No quality model adoption.	• Some quality activities were incorporated in the software development process.	• Software development process guided by a quality model.	• High quality model usage in the engineering department.	• Quality model completely adopted in the organization activities.
Software reuse measurement	• No metrics on level of reuse, payoff, or cost of reuse.	• Number of lines of reused code factored into cost models.	• Manual tracking of reuse occurrences of catalog parts.	• Analyses performed to identify expected payoffs from developing reusable parts.	• All system utilities, software tools, and accounting mechanisms instrumented to track reuse.
Systematic reuse process	• No reuse-based process.	• Some reuse activities were adopted in the development process. • Planning to adapt the software development process of the organization for a reuse-based process.	• Development process of the organization is adapted to reuse concepts.	• Reuse benefits and concepts are clear for the engineering team. • Development process is reuse-based.	• Systematic reuse process is enterprise-wide.
Origin of the reused assets	• No reuse assets.	• Build from scratch, some times indirectly.	• Build from existent products; adapting existing products.	• Build from existing products; extracted through a reengineering process.	• Planning the design and building of reusable assets according to product family.
Previous development of reusable assets	• No development of reusable assets.	• Parallel with development.	• Before development.	• Before development.	• Before development.

Figure 5. RiSE Maturity Model Levels: Processes Factors

5. Concluding remarks and Future Works

This paper describes the specification of the initial RiSE Maturity Model. It describes the approach for creating the model, its current structure and the levels it comprises. The results of the work on the RiSE project have been also taken into account during the development of the RiSE Maturity Model.

The RiSE Maturity Model is recommended to be used as a reference model in a Reuse Adoption Program, i.e. as a basis for estimating the level of software reuse practice within an organization. As future work, the RiSE Maturity Model aims at identifying the strengths of an organization with respect to software reuse and the opportunities for improvements. Such an analysis will serve as a catalyst for introducing reuse engineering and management practices and tools in a consistent manner, consolidating the strengths and favoring the understanding of the organization weak points.

Needless to say that the correct implementation of software reuse and the benefits for an organization adopting reuse in their processes can be evaluated only based on quantitative data. Therefore appropriate Reuse Business and Engineering metrics will be defined and are recommended to be used within the maturity model to measure the achievement of the respective objectives, the efficiency of the applied practices and the quality of the results obtained.

It is so hard to evaluate a model. To do this, we planning to apply our model in an industrial environment, at CESAR and Digital Port (`http://www.portodigital.org.br/`), to get more feedbacks from experts. We planning yet, to make the reuse model totally conformant with the ISO/IEC 12207 and ISO/IEC 15504 [ISO/IEC 1999].

The transition between one level to another for some factors still seems very subjective. Thus, another future work is the definition of specific guidelines to aid the organization to implement a reuse assessment (compliant with ISO/IEC 15504) to evaluate the current reuse practice stage and plan the next activities to implement the reuse adoption program. These specific question will help in organization reuse practices evolution.

An appraisal of the RiSE maturity level for one or more organizations (initially only small or medium organizations) will be performed to develop such guidelines and heuristics that help assess the maturity level of an organization.

Acknowledgment

This work is partially supported by Fundação de Amparo à Pesquisa do Estado da Bahia (FAPESB) - Brazil, process number: 674/2005; and *CompGov: Biblioteca Compartilhada de Componentes para E-gov*, MCT/FINEP/Ação Transversal process number: 1843/04.

References

Almeida, E. S. (2007). *RiDE: The RiSE Process for Domain Engineering*. Phd thesis, Federal University of Pernambuco (sandwich period at Universität Mannheim).

Almeida, E. S., Alvaro, A., Lucrédio, D., Garcia, V. C., and Meira, S. R. L. (2004). Rise project: Towards a robust framework for software reuse. In *IEEE International Conference on Information Reuse and Integration (IRI)*, pages 48–53, Las Vegas, USA. IEEE/CMS.

Almeida, E. S., Alvaro, A., Lucrédio, D., Garcia, V. C., and Meira, S. R. L. (2005). A survey on software reuse processes. In *IEEE International Conference on Information Reuse and Integration (IRI)*, pages 66– 71, Las Vegas, Nevada, USA. INSPEC Accession Number: 8689289 Digital Object Identifier: 10.1109/IRI-05.2005.1506451 Posted online: 2005-09-12 09:08:08.0.

Biggerstaff, T. J. and Richter, C. (1987). Reusability framework, assessment and directions. *IEEE Software*, 4(2):41–49.

Brito, K. S., Alvaro, A., Lucrédio, D., Almeida, E. S., and Meira, S. R. L. (2006). Software reuse: A brief overview of the brazilian industry's case. In *5th ACM-IEEE International Symposium on Empirical Software Engineering (ISESE), Short Paper*, Rio de Janeiro, Brazil. ACM Press.

Burkhard, N. (1993). Reuse-driven software processes guidebook. version 02.00.03. Technical Report SPC-92019, Software Productivity Consortium.

Card, D. N. and Comer, E. R. (1994). Why do so many reuse programs fail? *IEEE Software*, 11(5):114 – 115.

Chrissis, M. B., Konrad, M., and Shrum, S. (2004). *CMMI : Guidelines for Process Integration and Product Improvement*. Addison-Wesley Professional, 1st edition edition.

Davis, M. J. (1992). Stars reuse maturity model: Guidelines for reuse strategy formulation. In *Proceedings of the Fifth Workshop on Institutionalizing Software Reuse*, Palo Alto, California, USA.

Davis, T. (1993). The reuse capability model: A basis for improving an organization's reuse capability. In *Proceedings of 2nd ACM/IEEE International Workshop on Software Reusability*, pages 126–133. IEEE Computer Society Press / ACM Press.

Frakes, W. B. and Fox, C. J. (1995). Sixteen questions about software reuse. *Communications of the ACM*, 38(6):75–87. ACM Press. New York, NY, USA.

Frakes, W. B. and Isoda, S. (1994). Success factors of systematic software reuse. *IEEE Software*, 11(01):14–19.

Griss, M. L. (1994). Software reuse experience at hewlett-packard. In *16th International Conference on Software Engineering (ICSE)*, page 270, Sorrento, Italy. IEEE/CS Press.

Holibaugh, R., Cohen, S., Kang, K. C., and Peterson, S. (1989). Reuse: where to begin and why. In *TRI-Ada '89: Proceedings of the conference on Tri-Ada '89*, pages 266 – 277, Pittsburgh, Pennsylvania, United States. ACM Press.

ISO/IEC (1998). ISO/IEC 12207 software life cycle processes. International Standard 12207, ISO (the International Organization for Standardization) and IEC (the International Elechtrotechnical Commission).

ISO/IEC (1999). ISO/IEC 15504. information technology - process assesment part 1 - concepts and vocabulary (2004); part 2 - performing an assesment (2003); part 3 - guidance on performing an assesment (2004); part 4 - guidance on use for process improvement and process capability determination (2004); and part 5 - an exemplar process assesment model (1999). International Standard 15504, ISO (the International Organization for Standardization) and IEC (the International Elechtrotechnical Commission).

Koltun, P. and Hudson, A. (1991). A reuse maturity model. In *4th Annual Workshop on Software Reuse*, Hemdon, Virginia: Center for Innovative Technology.

Lim, W. C. (1994). Effects of reuse on quality, productivity and economics. *IEEE Software*, 11(5):23–30.

Lucrédio, D., Brito, K. S., Alvaro, A., Garcia, V. C., Almeida, E. S., Fortes, R. P. M., and Meira, S. R. L. (2007). Software reuse: The brazilian industry scenario. *submitted to the Journal of Systems and Software, Elsevier*.

Moore, M. M. (2001). Software reuse: Silver bullet? *IEEE Software*, 18(05):86.

Morisio, M., Ezran, M., and Tully, C. (2002). Success and failure factors in software reuse. *IEEE Transactions on Software Engineering*, 28(04):340–357.

Poulin, J. S. (2006). The business case for software reuse: Reuse metrics, economic models, organizational issues, and case studies. Tutorial notes.

Prieto-Díaz, R. (1991). Making software reuse work: An implementation model. *ACM SIGSOFT Software Engineering Notes*, 16(3):61–68.

Prieto-Díaz, R. (1993). Status report: Software reusability. *IEEE Software*, 10(3):61–66. IEEE Computer Society Press. Los Alamitos, CA, USA.

Rine, D. C. (1997a). Success factors for software reuse that are applicable across domains and businesses. In *ACM Symposium on Applied Computing*, pages 182–186, San Jose, California, USA. ACM Press.

Rine, D. C. (1997b). Supporting reuse with object technology - guest editor's introduction. *IEEE Computer*, 30(10):43–45.

Rine, D. C. and Nada, N. (2000a). An empirical study of a software reuse reference model. *Information and Software Technology*, 42(1):47–65.

Rine, D. C. and Nada, N. (2000b). Three empirical studies of a software reuse reference model. *Software: Practice and Experience*, 30(6):685–722.

Rine, D. C. and Sonnemann, R. M. (1998). Investiments in reusable software. a study of software reuse investiment success factors. *The Journal of Systems and Software*, 41:17–32.

SOFTEX (2007). Mps.br official web site (hosted by association for promoting the brazilian software excellence - softex).

SPC (1993). Reuse adoption guidebook, version 02.00.05. Technical Report SPC-92051-CMC, Software Productivity Consortium.

Tracz, W. (1990). Where does reuse start? *ACM SIGSOFT Software Engineering Notes*, 15(2):42 – 46.

Wartik, S. and Davis, T. (1999). A phased reuse adoption model. *The Journal of Systems and Software*, 46:13–23.