UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE EDUCAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM EDUCAÇÃO MATEMÁTICA E TECNOLÓGICA
CURSO DE MESTRADO

RICARDO TIBURCIO DOS SANTOS

PROCESSO DE DESENVOLVIMENTO DE SOFTWARE EDUCATIVO: UM ESTUDO DA PROTOTIPAÇÃO DE UM SOFTWARE PARA O ENSINO DE FUNÇÃO

RECIFE
2016

RICARDO TIBURCIO DOS SANTOS

PROCESSO DE DESENVOLVIMENTO DE SOFTWARE EDUCATIVO: UM ESTUDO DA PROTOTIPAÇÃO DE UM SOFTWARE PARA O ENSINO DE FUNÇÃO

Dissertação apresentada ao Programa de Pós-Graduação em Educação Matemática e Tecnológica, como requisito parcial para obtenção do título de Mestre em Educação Matemática e Tecnológica.

Orientador: Prof. Dr. Franck Gilbert René Bellemain.

RECIFE

2016

Catalogação na fonte
Bibliotecária Andréia Alcântara, CRB-4/1460

S237p Santos, Ricardo Tibúrcio dos.
Processo de desenvolvimento de software educativo: um estudo da prototipação de um software para o ensino de função / Ricardo Tibúrcio dos Santos. – Recife: O autor, 2016.
110 f. ; 30 cm.
Orientador: Franck Gilbert René Bellemain.
Dissertação (Mestrado) - Universidade Federal de Pernambuco, CE. Programa de Pós-graduação em Educação Matemática e Tecnológica, 2016.
Inclui Referências e Apêndices.

1. Matemática - Estudo e ensino. 2. Engenharia de software. 3. Software - Desenvolvimento. 4. UFPE - Pós-graduação. I. Bellemain, Franck Gilbert René. II. Título.

372.7 CDD (22. ed.) UFPE (CE2016-13)

RICARDO TIBURCIO DOS SANTOS

PROCESSO DE DESENVOLVIMENTO DE SOFTWARE EDUCATIVO: UM ESTUDO DA PROTOTIPAÇÃO DE UM SOFTWARE PARA O ENSINO DE FUNÇÃO

Dissertação apresentada ao Programa de Pós-Graduação em Educação Matemática e Tecnológica da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de mestre em Educação Matemática e Tecnológica.

Aprovado em: 15/02/2016.

BANCA EXAMINADORA

_____

Prof. Dr. Franck Gilbert René Bellemain (Orientador e Presidente)
Universidade Federal de Pernambuco

_____

Profa. Dra. Verônica Gitirana Gomes Ferreira (Examinadora Interna)
Universidade Federal de Pernambuco

_____

Prof. Dr. Rony Cláudio de Oliveira Freitas (Examinador Externo)
Instituto Federal do Espírito Santo

Dedico este trabalho a Maura Helena, mulher doce e guerreira que tenho todos os dias o orgulho de chamá-la de Mãe.

AGRADECIMENTOS

Agradeço a Deus pela oportunidade de cursar e concluir meu Mestrado.

A minha família, especialmente, a minha mãe por todo apoio e compreensão durante a escrita da dissertação.

Ao meu orientador, Prof. Dr. Franck Bellemain, pela confiança em mim depositada e todo apoio a mim dispensado.

À Banca Examinadora, constituída pelo Prof. Dr. Rony Freitas, que gentilmente apresentou sugestões significativas a esta pesquisa e, à Profa. Dra. Verônica Gitirana, que acompanhou o desenvolvimento desse trabalho, compartilhando generosamente conhecimentos e dando contribuições relevantes para a minha formação pessoal e profissional.

Ao Grupo de Pesquisa LEMATEC – Laboratório de Educação Matemática e Tecnológica, pelo apoio e disposição em participar efetivamente desta investigação.

E, por último, mas não menos importante, agradeço aos meus amigos, pelo companheirismo e apoio nos bons e maus momentos.

RESUMO

Esta pesquisa teve por objetivo construir, analisar e validar um Processo de Desenvolvimento de micromundo para a aprendizagem da Matemática, integrando os métodos da engenharia de requisitos e Ágeis, como elementos da Engenharia de Softwares, integrados com princípios da Engenharia Didática numa perspectiva transdisciplinar de exploração das potencialidades teóricas e tecnológicas dessas engenharias. Estudos atuais indicam que muitos recursos tecnológicos disponíveis são desenvolvidos centrados nas possibilidades oriundas da tecnologia, ou apenas nas teorias sobre a aprendizagem dos conhecimentos. A articulação entre contribuições teóricas e tecnologias ainda não é comumente realizada, e nos casos em que é feita, precisa ainda de sistematizações como processo de Engenharia. A partir dessas reflexões o quadro teórico-metodológico desta investigação foi constituído pelas Engenharias Didática e de Software, observando as contribuições de ambas para a construção de um processo de desenvolvimento de software educativo. A metodologia compôs-se da concepção, criação e análise de um processo de desenvolvimento de software educativo que articula potencialidades tecnológicas a teorias de ensino e aprendizagem. Uma equipe de colaboradores, composta por especialistas em diversas áreas da Matemática, foi formada para auxiliar na concepção, desenvolvimento e aperfeiçoamento do processo de software discutido neste estudo. Em um estudo de caso, validamos o processo criado desenvolvendo em colaboração com outros pesquisadores um software sobre a taxa de variação de funções matemáticas. Com esse experimento foram identificadas lacunas no processo concebido e aperfeiçoamentos foram realizados validando o processo de software criado. Os resultados deste estudo mostram que a integração de conhecimentos teóricos sobre o ensino e a aprendizagem de conhecimentos matemáticos com as potencialidades tecnológicas atuais é um fator a ser considerado para o desenvolvimento de softwares educativos, visto que o produto desenvolvido com essa perspectiva de articulação pôde contribuir para especificação e implementação dos requisitos alcançando as especificidades dos conhecimentos em questão.

Palavras-chave: Processo de desenvolvimento de software. Engenharia Didática. Engenharia de requisitos. Engenharia de Software Educativo.

ABSTRACT

This research aimed to build, analyze and validate a development process of microworlds for learning mathematics, integrating the methods of requirements engineering and Agile, as elements of software engineering, integrated to principles of Didactic Engineering a transdisciplinary perspective of exploitation theoretical and technological potentiality of these engineering. Current studies indicate that many available technological resources are developed focusing on opportunities arising from the technology, or only on theories about learning knowledge. The link between theoretical contributions and technologies is not commonly performed, and where it is done, it still need systematizations as engineering process. From these reflections the theoretical and methodological framework of this research was made up of Didactic and Software Engineering, noting the contributions of both to build an educational software development process. The methodology consisted of the design, creation and analysis of an educational software development process which combines the technological potential of teaching and learning theories. A team of employees, comprised of experts in various areas of mathematics, was formed to assist in the design, development and improvement of the software process discussed in this study. In a case study, we validate the process created developing, in collaboration with others researchers, a software on the rate of change of mathematical functions. In this experiment, gaps were identified in the process designed and improvements were carried out validating the process of software development created. The results of this study show that the integration of theoretical knowledge about teaching and learning of mathematical knowledge with current technological potential is a factor to be considered for the development of educational software, as the product developed with this perspective of articulation could contribute for specification and implementation of the requirements reaching the specifics of knowledge in question.

Keywords: Software Development Process. Didactic Engineering. Requirements Engineering. Educational Software Engineering.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

No cenário da sociedade atual é importante observar que as alterações da cultura são influenciadas pelo avanço da tecnologia: novas formas de trabalho, de relacionamento, pessoal e profissional, de tratamento de doenças, de ensinar e aprender, entre outras, modificam-se com a utilização de inventos tecnológicos, vivemos a cibercultura (LEMOS, 2003). Diante da crescente evolução das tecnologias, o ensino e a aprendizagem de conhecimentos passam, atualmente, por uma reconfiguração para acompanhar o desenvolvimento das tecnologias de informação e comunicação.

Compreendendo as possibilidades que os recursos tecnológicos viabilizam para as relações de ensino e aprendizagem, documentos oficiais, como os Parâmetros Curriculares Nacionais (BRASIL, 1998a), Orientações Curriculares para o Ensino Médio – OCEM (BRASIL, 2006) e Parâmetros Curriculares Nacionais para o Ensino Médio (BRASIL, 1998) apoiam a integração de novas tecnologias da informação e comunicação – NTIC, à prática docente. Especificamente para a Matemática, as OCEM trazem à tona a questão da escolha dos softwares a serem utilizados nas salas de aula. De acordo com o documento, "No uso de tecnologia para o aprendizado da Matemática, a escolha de um programa torna-se um fator que determina a qualidade do aprendizado" (BRASIL, 2006, p. 89).

Contudo, diante destes avanços, nos deparamos com duas questões a serem consideradas: 1. A utilização de novas tecnologias por si só não é suficiente para uma melhora significativa do ensino e da aprendizagem de conhecimentos; 2. O desenvolvimento de softwares educativos deve considerar, além dos aportes tecnológicos, elementos oriundos das reflexões sobre o ensino e sobre o ensino com tecnologias.

A articulação entre as questões levantadas é fundamentada na utilização e no desenvolvimento de recursos tecnológicos no ensino e aprendizagem de conhecimentos matemáticos. Algumas pesquisas (BENITTI; SCHLINDWEIN; SEARA, 2005; SANTOS, 2009) apresentam a problemática da qualidade dos softwares que estão sendo comercializados. Para esses pesquisadores, falta uma engenharia de *software* que dê suporte para a construção adequada dos programas. Nessas pesquisas foi observada a necessidade de articulação entre as diversas áreas de conhecimento envolvidas na concepção de software educativo – SE.

De acordo com Benniti et al (2005) "os softwares educacionais existentes – em sua grande maioria – possuem problemas que dificultam a sua utilização, dentre eles a falta de

uma base pedagógica que fundamente a sua construção" (p. 2). Com isso, compreendemos a base pedagógica, citada pelos autores, como a necessidade de inserir na concepção do *software* conhecimentos sobre o ensino e a aprendizagem dos conhecimentos que serão trabalhados. Para Bellemain et al (2014)

> Enunciar a necessidade da articulação entre soluções tecnológicas e abordagem teórica pode parecer uma evidência, porém muitas soluções tecnológicas para o ensino não se apoiam em tal princípio. Ora são essencialmente tecnocêntricas, ou seja, buscam explorar a potencialidade da tecnologia, mas desconsideram a produção acadêmica (disciplinar e didática) sobre os conteúdos veiculados pela tecnologia, ora se apoiam nos conhecimentos sobre os conteúdos e sua didática, mas exploram muito pouco das potencialidades do computador. (BELLEMAIN et al, 2014, p. 4).

As reflexões sobre o ensino com tecnologias e sobre a concepção de recursos que discutam a integração de estudos teóricos com as potencialidades tecnológicas é uma evidência nas pesquisas da área (BENITTI; SCHLINDWEIN; SEARA, 2005; SANTOS, 2009, ABREU et al, 2012; LIMA *et al*,2012, COSTA, 2012). Além disso, observamos nestes estudos que o auxílio aos objetivos de aprendizagem, oferecido pelos softwares educativos, é apresentado como baixo devido ao processo de concepção e construção dos mesmos.

Com a análise dos estudos, percebemos que é necessário considerar na engenharia dos SE as reflexões sobre o ensino e a aprendizagem com recursos tecnológicos. A partir destas observações, percebe-se que é necessária uma engenharia que contemple necessidades apresentadas nas pesquisas da Educação Matemática – EM, aliada a um estudo de potencialidades tecnológicas, pois ao pensarmos no desenvolvimento de softwares educativos, é importante considerar os resultados das pesquisas sobre o ensino e a aprendizagem.

1.1 Perspectiva coletiva de desenvolvimento

Os padrões de pesquisa na EM contemplam, em sua essência, características e particularidades do ensino e aprendizagem de conhecimentos. Segundo Garcia e Machado (2007) o objeto de estudo da Educação Matemática "é a compreensão, a interpretação e a descrição de fenômenos referentes ao ensino e à aprendizagem da matemática, nos diversos níveis de escolaridade, tanto na sua dimensão teórica, quanto prática" (p. 1). Ao compreender o objeto de estudo desta área, percebemos que as pesquisas de análise e desenvolvimento de softwares educativos não estão totalmente inseridas nela, pois existem características oriundas

da tecnologia que necessitam de fundamentação teórica e prática proveniente da Engenharia de Software – ES. Com isso, verificamos a necessidade de considerar contribuições teóricas e metodológicas da EM bem como da Engenharia de Softwares para a metodologia dessa pesquisa, a fim de possibilitar a concepção de um processo de desenvolvimento software educativo que considere as contribuições conjuntas da EM e da ES.

A presente investigação surgiu de estudos e discussões do Grupo de Pesquisa LEMATEC – Laboratório de Ensino da Matemática e Tecnologia. Esse laboratório é um grupo de pesquisa registrado no Diretório dos Grupos de Pesquisa no Brasil e seu atual cerne de investigação é a crescente introdução das tecnologias computacionais no ensino da Matemática tanto na modalidade presencial como a distância. As questões abordadas no grupo estão relacionadas com: o desenvolvimento e a avaliação de tecnologias, as condições de integração das tecnologias computacionais no ensino da matemática e, as contribuições e aos limites dessa integração e das modificações provocadas pela integração nas aprendizagens e no funcionamento do sistema didático. Os estudos do grupo fundamentam-se teoricamente na CSCL, Engenharia de software, na Epistemologia e Didática da Matemática (PORTAL DO LEMATEC, 2015).

No LEMATEC, desenvolve-se, atualmente, uma investigação com foco no estudo do desenvolvimento de um recurso tecnológico para o ensino e a aprendizagem de funções matemáticas. Neste estudo, pretende-se criar um recurso tecnológico para ser utilizado com estudantes de cursos de Cálculo, especificamente no conteúdo de taxa de variação, com a finalidade de verificar as contribuições do construto tecnológico para a aprendizagem dos conhecimentos em questão (SILVA, 2014).

Diante dos estudos em desenvolvimento no LEMATEC, ao percebermos convergências entre os objetivos das duas pesquisas: o presente estudo com a finalidade de desenvolver e analisar um processo de software educativo para conhecimentos matemáticos e o outro estudo com a intenção de conceber um software para a aprendizagem de funções; conseguimos direcionar os estudos para um ponto comum. Criando interações, a fim de verificar as possibilidades de relacionar os objetivos das pesquisas, inúmeras reuniões (online e presenciais), formulários, entrevistas, questionamentos etc., chegamos a percepção de que era possível obter resultados com a associação das pesquisas.

Nas discussões promovidas nos encontros do grupo de pesquisa supracitado, percebeu-se que existia uma convergência entre os interesses de investigação, no momento em que existe o interesse na análise da utilização das tecnologias digitais no ensino e

aprendizagem da Matemática. Com isso, articulamos as pesquisas para que uma fornecesse elementos de análise para a outra simultaneamente.

Criamos situações com as quais fosse possível promover as articulações necessárias para as investigações. Definimos a necessidade de reuniões periódicas (online e presenciais), utilização de ferramentas colaborativas de edição de textos, planilhas, desenhos, etc. (viabilizadas pelo Google Drive) para, em um momento inicial, compreender a proposta do software que se pretendia desenvolver. O presente estudo fornecia subsídios para o desenvolvimento do software pretendido obtendo da outra investigação os requisitos iniciais, as situações de uso, entre outras situações que serão detalhadas no quarto capítulo deste texto.

Especificamente para o estudo das funções matemáticas, Silva (2014) observou a necessidade de artefatos tecnológicos que contemplem as características exigidas por esse conhecimento, segundo ele,

> Dada a característica essencialmente dinâmica das funções reais de variáveis reais, as novas tecnologias computacionais têm se mostrado como aliadas para uma abordagem mais propícia de conceitos como taxa de variação (CASTRO FILHO, 2001; VILLA-OCHOA, 2011). Entretanto, a forma como função e seus aspectos têm sido abordados na escola se apoia principalmente em ferramentas estáticas, o que leva a uma percepção como relação entre pontos, sem estudar como a alteração nas variáveis de entrada afetam as relações entre os pontos correspondentes. (SILVA, 2014, p. 2).

Na perspectiva do ensino e aprendizagem de funções matemáticas, acreditamos que artefatos do tipo micromundo e/ou simulação configuram-se como recursos úteis para o desenvolvimento dos conhecimentos desta área de conhecimentos.

De acordo com Noss e Hoyles (1996, *apud* BELLEMAIN, 2002), "o termo micromundo foi inicialmente usado para definir um sistema que permite ***simular ou reproduzir*** um domínio do mundo real, e que tem como objetivo abordar e resolver uma classe de problemas" (p. 54, grifo nosso). As simulações e reproduções que os micromundos são capazes de produzir fornecem respostas aos resultados das pesquisas observadas nas discussões realizadas no Grupo de Pesquisa LEMATEC, para Silva (2014),

> Ao analisar o resultado de pesquisas que tratam das percepções dos alunos sobre o aspecto da variação em gráficos de funções, Gomes Ferreira (1997) conclui que uma das dificuldades dos alunos é de interpretar esses gráficos numa perspectiva variacional. (SILVA, 2014, p. 2)

É importante destacar que apenas a utilização do micromundo não é suficiente para garantir a aprendizagem. Segundo Almouloud (2007), para que a aprendizagem ocorra, "o micromundo deve ser incluído num dispositivo didático. São as características desse ambiente que garantirão a aprendizagem esperada" (p. 11). Entretanto, os ambientes micromundos/simulações configuram-se como elementos importantes para a aprendizagem de conhecimentos porque permitem ao aluno vivenciar uma prática matemática no contexto computacional. Nesse sentido, eles intervêm como recursos e integram um EIAH: "Existe então EIAH quando o sistema amplo (o artefato informático, a situação pedagógica criada, os diversos atores e seus papéis) é considerado" (TCHOUNIKINE, 2004, p.2, *tradução nossa*).

Ao delimitar o tipo de software (micromundo/simulação) concordamos com Tchounikine (2009) que existem engenharias específicas para um tipo de objeto X, onde X = situação de aprendizagem à distância, X = cenário para a aprendizagem colaborativa, X = simulação, X = geometria dinâmica, etc. Trata-se de um X de diversas naturezas, e os elementos de engenharia correspondentes igualmente (BELLEMAIN et al, 2015).

Nesse sentido, pretendemos, com o processo de software que será concebido, analisar, em um estudo de caso, o desenvolvimento de uma engenharia de software educativo que permita a construção de ambientes micromundo.

1.2 Características da proposta de Engenharia de Software Educativo

Sobre a Engenharia de *Software* Educativo – ESE, observamos um primeiro questionamento norteador que direcionou o presente estudo,

> O que dizer, então, da engenharia de software educativo (ESE)? Será apenas uma mudança no campo para o qual o software é elaborado, ou envolve também uma mudança estrutural na maneira de conceber e desenvolver as soluções, o que faz dele um campo de pesquisa e praxe privilegiado? (GALVIS-PANQUEVA, 1997, p. 18).

Com as indagações feitas, refletimos e procuramos respostas sobre os aspectos complexos da natureza das pesquisas na ESE. Segundo Galvis-Panqueva (1997), a ESE é um "novo campo, de natureza interdisciplinar, em que não basta saber fazer software com orientação educativa, ou ter brilhantes ideias educativas e dar-lhes uma sustentação informatizada" (p. 11).

Entretanto, Tchounikine (2004) considera que a ESE deve considerar uma perspectiva *transdisciplinar* para destacar o fato que os conhecimentos das áreas envolvidas não sejam apenas reunidos, mas sejam integrados, numa perspectiva de interseção para viabilizar a criação de novos conhecimentos. Segundo o autor, "o desenvolvimento de um EIAH é um ecótono campo científico na interseção de diferentes disciplinas, o que representa um conjunto de desafios únicos. É necessário abordar esse campo específico com uma perspectiva transdisciplinar" (p. 17, *tradução nossa*).

Assim, compreendemos que se torna necessário à integração de várias áreas de pesquisa para que o software que se pretende conceber atenda as diferentes necessidades do ensino e da aprendizagem dos conhecimentos envolvidos, sejam elas cognitivas, didáticas, epistemológicas, tecnológicas e de naturezas diversas. Compreendemos como ESE, o processo de idealização e desenvolvimento de interfaces ou artefatos com a finalidade de contribuir para a melhoria das relações de ensino e aprendizagem de áreas distintas do conhecimento.

Em uma perspectiva prática de desenvolvimento de *software*, concordamos com Costa e Costa (2013), quando observam a importância da utilização de metodologias definidas, segundo eles: "A necessidade de definir uma metodologia de desenvolvimento de software educativo tem como propósito minimizar/reduzir erros durante o processo de desenvolvimento e garantir a qualidade do recurso em si" (p. 4). Sobre essas metodologias, os autores acreditam que os *métodos ágeis,*

> [...] procuram fornecer e dar prioridade aos novos requisitos do software e avaliar as iterações do mesmo. Dão enfoque ao papel das pessoas, sendo que as competências da equipe de desenvolvimento devem ser reconhecidas e exploradas. Os elementos da equipe são livres para utilizar os seus próprios métodos de trabalho sem serem prescritos processos (COSTA E COSTA, 2013, p. 3-4).

As metodologias ágeis foram desenvolvidas como uma alternativa aos modelos prescritivos e de processo de desenvolvimento de software. Mesmo sendo uma alternativa aos modelos tradicionais elas não deixam de ter *princípios a serem seguidos* que foram enunciados num manifesto: o Manifesto Ágil (2001) (que será abordado nos próximos capítulos). Acreditamos que os métodos ágeis podem contribuir para o desenvolvimento de SE na medida em que seus princípios podem nortear a gestão das atividades, da organização da equipe e do desenvolvimento dos produtos.

Com o discutido até aqui, partimos da hipótese que uma ESE que alie os estudos da Didática da Matemática com as potencialidades tecnológicas promoverá a criação de softwares educativos que atendam as necessidades específicas dos conhecimentos envolvidos, pois, além do indicativo apresentado nas pesquisas sobre Tecnologia e Educação Matemática, acreditamos que a produção de conhecimentos na área da Didática da Matemática aliada com os estudos da Engenharia de Software pode contribuir para a criação de recursos que auxiliem a aprendizagem de conhecimentos matemáticos.

A escolha pela Didática da Matemática justifica-se por encontrarmos nessa área de estudo pesquisas teóricas e experimentais, teorias, hipóteses e verificações sobre o ensino e a aprendizagem da Matemática que contribuem para a observação de características dos conhecimentos matemáticos que devem ser contempladas em softwares educativos.

1.3 Objetivos da pesquisa

A partir do observado na problemática da articulação das potencialidades dos softwares com as pesquisas na área da Didática da Matemática, configura-se como objetivo geral deste estudo:

Construir, analisar e validar um Processo de Desenvolvimento de micromundo para a aprendizagem da Matemática, integrando os métodos da engenharia de requisitos e Ágeis, como elementos da Engenharia de Softwares, integrados com princípios da Engenharia Didática numa perspectiva transdisciplinar de exploração das potencialidades teóricas e tecnológicas dessas engenharias.

E objetivos específicos:

- Analisar metodologias de desenvolvimento de softwares educativos existentes, identificando potencialidades e limitações;

- Articular elementos das teorias da Didática da Matemática e da Engenharia de software para desenvolver um processo de Engenharia de Software;

- Estudar um caso de desenvolvimento de um software seguindo o processo criado.

1.4 Estrutura do texto

No segundo capítulo apresentamos uma revisão de literatura sobre o atual cenário do desenvolvimento de softwares educativos e quais são as contribuições dos trabalhos analisados para o desenvolvimento de softwares no âmbito do ensino e da aprendizagem da Matemática. No terceiro capítulo discutimos as teorias que fundamentam a presente pesquisa fazendo uma articulação entre elas e observando a contribuição de cada uma para esse trabalho. Ainda nesse capítulo apresentamos a abordagem da Engenharia Didático-Informática – EDI, como procedimento teórico e metodológico para a concepção e desenvolvimento de softwares educativos.

No quarto capítulo encontra-se o percurso metodológico adotado para alcançar os objetivos propostos. Apresentamos neste capítulo todos os processos realizados até a concepção da versão final do processo de desenvolvimento de SE discutido nesta pesquisa. No quinto capítulo apresentamos como foi utilizado o Processo de Desenvolvimento de Software Educativo no cenário do Grupo de Pesquisa LEMATEC. Discutimos nesse capítulo como ocorreu o lançamento da versão inicial do software bem como as implementações necessárias após a etapa de testes.

Reservamos o último capítulo para a análise e discussão dos resultados e as considerações finais sobre o estudo desenvolvido. Esperamos que esse trabalho possa enriquecer o cenário de desenvolvimento de softwares educativos e motivar outros estudos na mesma temática.

2 PANORAMA GERAL DO DESENVOLVIMENTO DE SOFTWARE EDUCATIVO

Nesse capítulo apresenta-se uma revisão de literatura sobre o atual cenário do desenvolvimento de softwares educativos, quais são os processos, os métodos, ferramentas, entre outros recursos que são utilizadas para a concepção de um SE. Nas próximas páginas, respondemos alguns questionamentos que surgiram durante a análise: Quais resultados são obtidos a partir das atuais metodologias de desenvolvimento de software educativo? Quais são as orientações e lacunas das pesquisas para o desenvolvimento de softwares educativos?

2.1 Recurso a revisões sistemáticas no desenvolvimento de softwares educativos

A tradicional revisão de literatura, realizada em grande parte das pesquisas em Educação Matemática e Tecnológica, além de outras áreas, possui elementos que contribuem de forma significativa para o desenvolvimento de um estudo. Entretanto, uma abordagem de revisão que difere em sistematização e refinamento de busca está sendo utilizada e por ser tão específica iremos analisar trabalhos que recorreram a tal metodologia. Essa revisão é chamada de *revisão sistemática de literatura* ou apenas *revisão sistemática* e pode ser definida, de acordo com Kitchenham e Chaters (2007), do seguinte modo,

> Uma revisão sistemática de literatura (muitas vezes referida como revisão sistemática) é um meio de identificar, avaliar e interpretar **todas as pesquisas disponíveis** relevantes para uma determinada questão de pesquisa ou área de tópico, ou fenômeno de interesse (p. 11, 2007, *tradução nossa, grifo nosso*).

Observando o termo destacado na definição percebemos que a revisão sistemática é um procedimento que demanda tempo e técnicas de busca bem definidas, mas que traz um vasto panorama sobre o que se pretende pesquisar. As vantagens da utilização desse método podem ser resumidas nos seguintes itens: 1. A metodologia bem definida dessa revisão torna menos provável que os resultados sejam tendenciosos; 2. Os resultados podem fornecer informações sobre os efeitos de algum fenômeno através de uma ampla gama de configurações e métodos empíricos. 3. No caso de estudos quantitativos, é possível combinar os dados usando meta-técnicas analíticas. Isso aumenta a probabilidade de detectar efeitos reais que estudos menores individuais são incapazes de detectar. (KITCHENHAM & CHATERS, 2007).

Cook et al. (1997) *apud* Conforto et al. (2011) confirma que a revisão sistemática é baseada na aplicação de métodos científicos rigorosos, podendo alcançar melhores resultados e reduzir erros e a influência do pesquisador responsável pela investigação. “Esse processo permite ao pesquisador compilar dados, refinar hipóteses, estimar tamanho de amostras, definir melhor o método de pesquisa a ser adotado para aquele problema, e por fim definir direções para futuras pesquisas”. (CONFORTO et al, 2011, p. 2).

O método da revisão sistemática tem origem na medicina baseada em evidências, mas a abordagem apresentada pelos autores está voltada para a engenharia de softwares, pois, segundo eles,

> [...] a pesquisa de engenharia de software tem relativamente pouca pesquisa empírica em comparação com o domínio médico; métodos de pesquisa utilizados por engenheiros de software não são tão rigorosos quanto os geralmente usados por pesquisadores médicos; e a quantidade de dados empíricos existentes em engenharia de software é privada (KITCHENHAM & CHATERS, 2007, p. 9, *tradução nossa*).

Com o exposto, pretendemos analisar pesquisas na área de engenharia de softwares educativos, que utilizaram a metodologia da revisão sistemática de literatura, com a finalidade de compreender o panorama atual da produção de SE, conhecendo pesquisas relevantes na área para analisar as sugestões de tais investigações servindo de diagnóstico inicial para a construção do modelo de processo de software proposto nesta pesquisa.

2.2 Observações nas revisões sistemáticas sobre o desenvolvimento de software educativo

Com a intenção de fornecer subsídios para pesquisas na área da engenharia de software educativo, Abreu *et al* (2012) sugere que o resultado de sua pesquisa "gera a fundamentação técnica pedagógica para, em um passo seguinte, o desenvolvimento ou aprimoramento de uma metodologia de desenvolvimento de software educacional". (p. 1). Para ele "independente da classificação do software, é necessário que seja produzido levando em conta não só conceitos de engenharia de software, mas as teorias pedagógicas inerentes ao contexto educacional" (p. 1). Com efeito, é possível observar que a problemática da presente investigação está em consonância com a pesquisa aqui discutida, pois o autor acredita que se faz necessário aliar os resultados e as teorias das pesquisas sobre o ensino e a aprendizagem dos conhecimentos para o desenvolvimento dos softwares educativos.

A partir de sua problematização, com a finalidade de delimitar sua busca e alcançar os objetivos de sua pesquisa, Abreu *et al* (2012) formulou quatro questionamentos, são eles,

> Q1 – Quais métodos, técnicas e ferramentas existem para apoiar o desenvolvimento de software educacional? Q2 – Que aspectos pedagógicos são utilizados pelos métodos, técnicas, e ferramentas que apoiam o desenvolvimento do software educacional? Q3 – Quais são as questões abordadas pelos métodos, técnicas e ferramentas utilizados no desenvolvimento do software educacional? Q4 – Quais os problemas em aberto segundo os métodos técnicas e ferramentas do processo desenvolvimento de software educacional? (ABREU et al, 2012, p. 3).

O processo de busca deu-se através de procedimentos automatizados e manuais. A *string* de busca resultante dos estudos iniciais foi: “("software engineering") AND (method OR technique OR process OR tool OR guideline OR framework) AND ("educational software" OR “learning software” OR “educational games” OR “educational simulators” OR tutorials OR “educational Programming Languages” OR “educational multimedia”)”. As bibliotecas digitais utilizadas no processo de busca automatizada foram: IEEE Computer Society Digital Library, ACM, Science Direct, Scopus, El compendex, Web Of Science. As conferências utilizadas no processo de busca manual foram: International Conference on Informatics Education and Research (SIGED); Journal of Interactive Media in Education; Simpósio Brasileiro de Informática na Educação (SBIE); e Revista Brasileira de Informática na Educação. Foram analisados 1636 estudos inicialmente e 65 foram selecionados como relevantes para o processo de investigação.

Os principais resultados do estudo de Abreu *et al* (2012) ratificam a hipótese da presente pesquisa e são de fundamental importância para direcionar estudos na ESE. Dos quatro questionamentos propostos, observamos contribuições significativas para a pesquisa em ESE, nos parágrafos seguintes apresentamos algumas considerações.

Em relação ao primeiro questionamento, podemos observar a Tabela 1 que apresenta quais tipos de tecnologias são utilizadas no desenvolvimento de software educacional (é válido ressaltar que o autor utiliza o termo “tecnologia” para se referir aos métodos, técnicas e ferramentas utilizadas para desenvolver softwares educativos). Das informações apresentadas é possível perceber que o Design (Design de projeto de arquitetura) é a forma mais utilizada para se desenvolver SE, já os Processos, foco da presente pesquisa, aparecem em sexto lugar.

Acreditamos que um possível motivo para o Design aparecer como primeiro colocado é o foco central na produção: a interface, considerando que o ensino de um conteúdo depende essencialmente da apresentação (multimídia). A definição de Design, apresentada pelos autores, está relacionada ao processo de definição da arquitetura, módulos, interfaces e outras características de um sistema quanto o resultado desse processo. Entretanto, o presente estudo não tomou como objetivo a verificação dessa hipótese.

Tabela 1 – Mecanismos para o desenvolvimento de software educacional, Abreu et al (2012).

Classificação	Porcentagem	Estudos Primários
Design	20,97%	EP08, EP13, EP15, EP47, EP48, EP49 EP50, EP53, EP55, EP57, EP58, EP60, EP61
Frameworks	16,13%	EP01, EP16, EP17, EP21, EP23, EP32, EP37, EP41, EP43, EP51
Ferramentas	12,9%	EP04, EP11, EP12, EP28, EP31, EP39, EP54, EP59
Métodos/Metodologia	12,9%	EP03, EP05, EP07, EP09, EP29, EP33, EP35, EP52
Modelo	11,29%	EP27, EP36, EP40, EP 42, EP44, EP45, EP65
Processos	8,06%	EP06, EP30, EP38, EP56, EP62
Arquitetura	6,45%	EP18, EP20, EP22, EP25
Guidelines	4,84%	EP19, EP24, EP65
Linguagens	3,23%	EP02, EP14
Componentes	1,61%	EP10
Técnicas	1,61%	EP34

Em relação aos aspectos pedagógicos, segundo questionamento proposto pelo autor, observamos que as atuais pesquisas procuram apoiar suas construções em teorias da aprendizagem, teorias cognitivas, aspectos interdisciplinares, etc., como pode ser verificado na Tabela 2.

Tabela 2 – Aspectos pedagógicos, Abreu et al (2012).

Classificação	Porcentagem	Estudos Primários
Aprendizagem Colaborativa	16,7%	EP05, EP12, EP15, EP16, EP27, EP61, EP62
Atividades	14,3%	EP04, EP16, EP33, EP38, EP56, EP57
Exploração do Currículo	14,3%	EP03, EP04, EP29, EP30, EP37, EP44
Construtivismo	11,9%	EP07, EP08, EP17, EP26, EP61
Multimídias	11,9%	EP23, EP24, EP38, EP48, EP49
Usabilidade	11,9%	EP03, EP09, EP31, EP38, EP55
Metodologia Interdisciplinar	9,5%	EP02, EP30, EP35, EP42
Exploração da Teoria	4,8%	EP01, EP42
Acompanhamento do Progresso do Aluno	4,8%	EP26, EP58

Com essa tabela, os autores ressaltam a importância do desenvolvimento de SE aliado a perspectivas pedagógicas, observou-se que 75,38% dos estudos analisados apresentaram aspectos pedagógicos para o desenvolvimento. De acordo com os autores,

> [...] hoje já existe uma preocupação significativa com a pedagogia de ensino quando se é adotado software educativo como recurso de auxílio neste processo. No entanto, o ideal é que em 100% dos softwares educacionais houvesse a preocupação com conceitos pedagógicos, uma vez que o software

> é utilizado para transmitir conteúdo e tais conceitos estão intimamente relacionados com a atividade de ensino (ABREU *et al*, 2012, p. 8).

Do questionamento Q4, referente às questões em aberto na área de desenvolvimento de software educacional, de acordo com o autor,

> [...] percebe-se que muito tem se discutido sobre tecnologias aplicadas no desenvolvimento de software educacional e sobre projeto de software. São áreas que possuem uma influência muito grande no desenvolvimento de software educacional. Dentre os temas abordados por estes estudos ainda aparece com bastante frequência o tema da usabilidade de software e da acessibilidade que são itens imprescindíveis para que os softwares educacionais, considerando que, sem pelo menos um destes itens bem planejado e executado não se faz software educacional (ABREU *et al*, 2012, p. 9).

Do trabalho analisado conclui-se que as atuais pesquisas apresentam a preocupação em articular a ESE a teorias sobre o ensino e a aprendizagem de conhecimentos. Além disso, o recurso a metodologias de desenvolvimento de *software* configura-se como importante aliado à qualidade dos softwares produzidos com finalidades pedagógicas. As sugestões e orientações são de fundamental importância para estudos na área da ESE.

Outra pesquisa, que também utilizou a revisão sistemática de literatura como metodologia, teve por objetivo obter informações sobre a evolução dos processos de desenvolvimento de SE. O estudo de Lima *et al* (2012) levantou as técnicas e os processos utilizados para o desenvolvimento de software educativo. Os autores acreditam que existe a necessidade de uma equipe multidisciplinar para o desenvolvimento e concepção desse tipo de software,

> Por ser voltado à educação, este tipo de software requer o envolvimento de profissionais das mais diversas áreas, como psicólogos, professores, especialistas na área de conhecimento, e além destes, profissionais na área de informática. É crescente a necessidade de softwares educativos no ensino. O desenvolvimento desses sistemas é um desafio para engenheiros de softwares que devem se preocupar com o lado pedagógico e cognitivo. (LIMA et al, 2012, p. 1).

O procedimento de busca, utilizado na pesquisa de Lima et al (2012), pelos trabalhos relacionados deu-se início com a formulação de uma pergunta de pesquisa primária: Após a discussão dos objetivos, chegou-se a questão de pesquisa: “What methods are used to develop educational software in the context of Software Engineering? (Quais métodos são usados para

desenvolver software educativo no contexto da Engenharia de Software?)" (p. 2). Partindo da questão de pesquisa a *string* de busca foi formulada: (Method OR Process OR tool OR technique) AND ("educational software" OR "educative software") AND ("software engineering"). As buscas automáticas foram realizadas nas seguintes bases de dados: ACM Digital Library e IEEE Xplore; Já as buscas manuais foram realizadas na Revista Brasileira de Informática na Educação e no Simpósio Brasileiro de Informática na Educação (SBIE).

Com os resultados das buscas realizadas 724 estudos foram encontrados. A partir deles foi realizada uma triagem e 27 estudos foram selecionados para análise. Dos 27 estudos, foi possível observar que o "Processo" foi o método mais utilizado para desenvolver software educativo, como pode ser observado na Figura 1. De acordo com Lima et al (2012), um processo de software pode ser definido como "um conjunto estruturado de atividades relacionadas, utilizadas para a produção de software" (p. 1) Definição que está em consonância com o referencial teórico pesquisado no presente estudo.

Figura 1 – Gráfico da quantidade de estudos por tópico pesquisado, LIMA et al, 2012.

Ambas as pesquisas aqui discutidas apresentam resultados importantes para o desenvolvimento de softwares educativos. Uma das contribuições observadas é a ampla utilização de metodologias padronizadas para o desenvolvimento de softwares. Acreditamos que os procedimentos que levam em consideração estratégias, técnicas, organização de uma equipe de trabalho, entre outros fatores que fazem parte de um procedimento de construção de softwares, são de fundamental importância para a qualidade do produto que será criado.

Outra contribuição observada está na adaptação de métodos de engenharia de *software* existentes para a construção de novos produtos. Essas adaptações, segundo Lima (2012), apresentam melhorias em relação aos processos já existentes. Não foi possível compreender,

no estudo analisado, quais os motivos das adaptações realizadas, mas acreditamos que as adaptações devem provir de observações feitas nos produtos criados, visando melhorias, com o objetivo de atender outras necessidades dos usuários, ou até os projetos anteriores que não foram eficazes.

Lima (2012) destaca também a importância de uma equipe pluridisciplinar para o desenvolvimento de SE, entretanto, é muito importante organizar o trabalho dessa equipe porque o fato de reuni-la não é suficiente para a concepção/desenvolvimento de produtos que atendam as necessidades específicas dos conhecimentos envolvidos. A articulação e integração entre a equipe e os conhecimentos devem ser promovidas a partir de uma metodologia de trabalho definida. Nesse contexto, nossa hipótese é que as metodologias ágeis, adaptadas ao contexto de software educativo, possam contribuir para organizar o trabalho da equipe e outros encaminhamentos do trabalho de desenvolvimento.

Por último, foi constatada a atual preocupação em integrar os conhecimentos da Engenharia de *Software* com as teorias da aprendizagem. Fato que ratifica a hipótese da presente pesquisa, ao passo que se torna de fundamental importância aliar os construtos teóricos sobre o ensino e a aprendizagem de conhecimentos aos pressupostos da Engenharia de *Software.*

Percebemos que a utilização de metodologias ágeis no desenvolvimento de softwares educativos ainda é pouco consolidada, de acordo com as revisões sistemáticas apresentadas. Com isso, na próxima sessão, apresentamos alguns trabalhos sobre o desenvolvimento de SE, utilizando metodologias ágeis, a fim de obter subsídios para a construção do modelo de processo discutido nessa pesquisa.

2.3 Paradigmas atuais do desenvolvimento de software educativo

Partindo da problemática do desenvolvimento de softwares educativos para o ensino da Matemática buscamos por pesquisas que versassem sobre tal temática com a finalidade de observar quais são os métodos e procedimentos utilizados para os softwares que tem por objetivo auxiliar o ensino e a aprendizagem da Matemática. Além disso, verificar as sugestões, orientações e o que ainda não foi pesquisado sobre o desenvolvimento desses softwares.

Costa e Costa (2013), com o objetivo de discutir uma proposta de desenvolvimento de SE, apresentam uma proposta de engenharia que tem por base o design centrado no utilizador – DCU, aliado à prática das metodologias ágeis. De acordo com os autores, o processo desenvolvido é "extremamente útil para garantir a qualidade dos pacotes de software educativo para a Matemática". (p. 1). Mesmo apresentando uma proposta para o desenvolvimento de softwares matemáticos, os autores não trabalharam com um conhecimento específico, a proposta apresentada é abrangente para os vários conhecimentos matemáticos.

Os autores indicam que os recursos tecnológicos contribuem para a aprendizagem de conhecimentos, porém quando analisam a inserção das novas tecnologias nos ambientes de ensino e aprendizagem questionam a qualidade dos softwares disponíveis, e acreditam que o processo de engenharia de SE é de fundamental importância para a qualidade dos softwares educativos, de acordo com eles, "o desenvolvimento de software educativo de qualidade implica uma avaliação formativa dos protótipos concebidos, pelas equipes, durante o processo de desenvolvimento" (p. 1) (LOUREIRO, 2002; COUTINHO; CHAVES, 2001; GOMES, 2000 apud COSTA E COSTA, 2013, *tradução nossa*).

Com isso, uma das conclusões observadas no trabalho de Costa e Costa (2013) ratifica a hipótese da presente pesquisa. De acordo com os autores, os procedimentos de desenvolvimento de SE aliados às teorias de aprendizagem colaboram com a qualidade dos produtos a serem desenvolvidos. Além disso, indicam que os processos não são capazes de resolver todos os problemas, mas pressupõe a qualidade dos produtos.

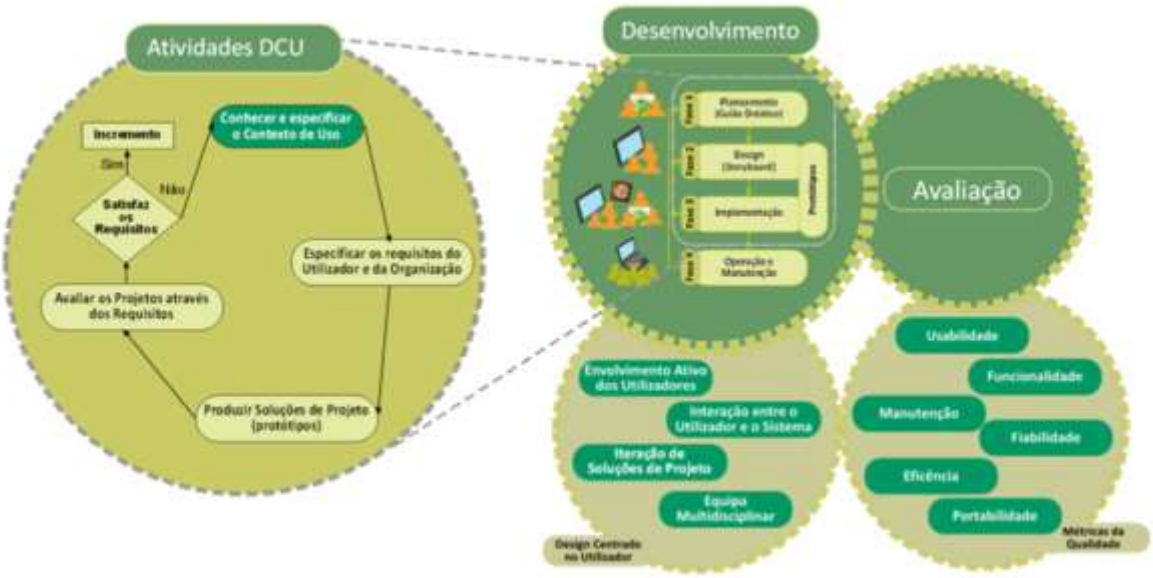
Em pesquisas anteriores, os autores acreditavam que formando uma equipe com profissionais técnicos (designers e programadores) e professores da área a que se destinava o software seria suficiente para construir produtos de qualidade. Porém, após o desenvolvimento de dois sistemas de software educativo, perceberam que existe certa volatilidade dos requisitos educacionais, tal situação é fundamentada na dificuldade de obter e quantificar os requisitos, além da necessidade de uma compreensão plena do conhecimento a ser auxiliado com o software (COSTA E COSTA, 2013).

Para superar o problema com os requisitos educacionais, os autores observam que a utilização do design centrado no utilizador serve "para descrever os processos de um projeto em que os utilizadores finais têm influência na forma como este é conduzido" (p. 5). Para eles, as contribuições do design centrado no utilizador estão relacionadas com a obtenção direta de informações pelo usuário final,

> Alguns métodos do Design Centrado no Utilizador sondam os utilizadores sobre as necessidades que estes possuem em determinada área educacional, envolvendo-os em partes específicas do processo de desenvolvimento. Por outro lado, existem métodos em que os utilizadores têm uma maior presença, integrando a equipe, isto é, são envolvidos como elementos durante todo o processo (ABRAS, MALONEY-KRICHMAR, PREECE, 2004 apud COSTA E COSTA, 2013, p. 5).

O produto das discussões e pesquisas apresentadas por Costa e Costa (2013) pode ser verificado na Figura 2. Segundo os autores, foi criada uma "Metodologia Híbrida de Desenvolvimento Centrado no Utilizador (COSTA, 2012), passível de ser utilizada no desenvolvimento de SE para a Matemática" (p. 7).

Figura 2 – Metodologia Híbrida de Desenvolvimento Centrado no Utilizador, Costa e Costa (2012).



Da pesquisa analisada algumas observações devem ser feitas para o direcionamento de estudos sobre o desenvolvimento de softwares e de modelos de desenvolvimento. Em relação à utilização do design centrado no utilizador, percebemos que a principal fonte de informações para a análise dos requisitos concentrou-se nos usuários, o que se configura como importante, mas acreditamos que para a concepção de SE a consulta a pesquisas teóricas sobre o ensino e a aprendizagem dos conhecimentos é indispensável.

Para um ambiente micromundo observa-se que os usuários são os pesquisadores, os professores e os estudantes. Os pesquisadores, no sentido do desenvolvimento, tornam-se usuários com a finalidade de propor e analisar situações de utilização do ambiente, entre

outros aspectos. Os professores, também são usuários na medida em que configuram os diversos ambientes e situações para a utilização do artefato inserindo o mesmo em alguma situação didática objetivando o ensino e a aprendizagem de algum conhecimento específico e os estudantes, por sua vez, utilizam-se do ambiente desenvolvido para aprender algum conhecimento.

Com as discussões realizadas até aqui inferimos que a perspectiva de desenvolvimento de ambientes micromundos tomando por base estudantes, dos diversos níveis, como utilizadores, seria prejudicada caso os requisitos do ambiente fossem adquiridos apenas com a opinião dos utilizadores. Dada a complexidade da configuração dos micromundos, não seria viável obter os requisitos através dos usuários finais, visto que estes ambientes necessitam de situações didáticas para efetivar o aprendizado do conhecimento a ser desenvolvido.

Ainda sobre os requisitos, existe um ramo específico da Engenharia de softwares para a obtenção dos mesmos, a engenharia de requisitos, que se configura como aporte útil para que se conheçam os requisitos de um software e até de um sistema (será abordada no próximo capítulo).

Outra pesquisa, também na perspectiva do desenvolvimento de SE utilizando metodologias ágeis, apresenta uma experiência de engenharia de objetos de aprendizagem. Lapolli et al (2009) consideram importante a utilização das metodologias de desenvolvimento com a finalidade de atender as necessidades organizacionais e pedagógicas, porém acredita que as metodologias tradicionais de desenvolvimento não satisfazem as necessidades das propostas atuais, segundo eles,

> A abordagem de desenvolvimento tradicional já não satisfaz as necessidades das propostas educacionais atuais, principalmente as de domínio complexo, pois nem sempre contribuem para a aprendizagem final. Portanto, é preciso pesquisar abordagens de desenvolvimento de recursos educacionais mais adequados a prática pedagógica (LAPOLLI et al, 2009, p. 250).

Discordamos dos autores quanto à utilização de abordagens tradicionais. Os autores consideram as primeiras metodologias para desenvolvimento como “tradicionais”, porém essas formas de desenvolvimento de software continuam sendo eficientes para a criação de diversos produtos. O que podemos perceber é que a utilização de metodologias mais antigas serve como base teórica e prática para os modelos mais recentes. Os métodos recentes são desenvolvidos baseados nos antigos, o que caracteriza uma evolução e não uma substituição.

Lapolli et al (2009) utilizaram a metodologia ágil, em específico o BDD Behaviour-Driven Development (desenvolvimento orientado a comportamento), um processo que engloba a análise de requisitos, até o desenvolvimento do código. De acordo com eles, o BDD é a "a união de várias práticas consideradas ágeis e úteis no desenvolvimento de software, cuja ênfase está nas funcionalidades de alto valor e na redução dos custos de mudança por meio da identificação do que de fato está sendo testado" (Lapolli et al, 2009, p. 253). Baseados nessa metodologia, os autores chegaram ao modelo apresentado na Figura 3.

Figura 3 – Modelo de projeto instrucional dirigido pelo comportamento, Lapolli et al (2009)

Em relação à obtenção de requisitos percebemos que a consulta sobre a aprendizagem dos conhecimentos técnicos foi realizada através de entrevistas. Observamos que documentos, teorias de aprendizagem, cognitivas ou outras, não foram consultados. Os autores definem sua busca da seguinte forma,

> A fim de identificar os requisitos e funcionalidades que deveriam ser abordados desenvolvidos nesse OA, realizamos entrevistas semi-estruturadas com instrutores e alunos no Instituto de Controle do Espaço Aéreo (ICEA), em setembro de 2007, onde apuramos as dificuldades em se construir cognitivamente um modelo do momento em que os conceitos abstratos de Meteorologia influenciam significativamente nos procedimentos operacionais. Na elaboração da proposta de atividades contamos com a colaboração de um controlador de tráfego aéreo com uma gama de conhecimentos, habilidades, convicções e conceitos adquiridos ao longo dos 20 anos de atividade profissional. (LAPOLLI et al, 2009, p. 257).

A identificação de requisitos por meio da consulta aos usuários é uma prática possibilitada pelas metodologias ágeis. Entretanto, para o desenvolvimento de softwares educativos, em específico para ambientes micromundos, faz-se necessário reunir requisitos

sobre ensino e aprendizagem. Analisando os estudantes como um dos usuários finais nesse tipo de ambiente, percebe-se a inviabilidade de obter requisitos didáticos ou cognitivos, por exemplo, visto que o aluno não possui acesso a teorias cognitivas ou didáticas (considerando alunos do ensino fundamental, por exemplo).

O usuário identificado nas metodologias ágeis, para o desenvolvimento de softwares sem intenção didática, é o cliente, esse deve ficar satisfeito com o produto. A perspectiva de usuário é diferente no caso de SE, visto que, como já fora observado, a pluralidade de usuários de um SE é bastante ampla. Nesse sentido, ao considerarmos alguns dos princípios ágeis para o processo de desenvolvimento de SE, desenvolvido neste estudo, consideramos a pluralidade da natureza dos usuários e a importância da participação dos mesmos na construção dos softwares.

Mesmo com a participação dos usuários no desenvolvimento de um SE, é de fundamental importância garantir que todos os requisitos necessários sejam identificados. Com isso, percebemos a necessidade da utilização da engenharia de requisitos para obter características, funcionalidades, especificações, etc. dos ambientes a serem desenvolvidos com o processo discutido neste estudo. Com esse método, é possível identificar de forma precisa as necessidades dos usuários e do sistema.

2.4 Equipe pluridisciplinar e metodologias ágeis de desenvolvimento

Percebemos nas pesquisas analisadas que a utilização das metodologias ágeis é complementada com outras possibilidades de desenvolvimento de software, porém para decidir qual metodologia utilizar para o desenvolvimento de SE encontramos em Costa (2012) as vantagens e desvantagens de algumas metodologias usuais de desenvolvimento de software, como pode ser verificado na Tabela 3.

Tabela 3 – Vantagens e desvantagens dos métodos de desenvolvimento de software, Costa (2012)

Designação	Vantagens	Desvantagens
Método em Cascata (anos 70)	Funciona bem para equipas tecnicamente mais fracas. É produzida documentação em cada fase.	Estrutura rígida e procedimentos inflexíveis. Não reconhece a necessidade de retornar às fases anteriores e corrigir erros. Versão operacional do sistema apenas disponível numa fase avançada.
Método Prototipagem (anos 80)	Apresenta resultados sem necessitar de toda a informação no início do projeto. É útil quando os requisitos mudam rapidamente e o utilizador está relutante em aceitar um conjunto de requisitos. Ajudam a definir os requisitos.	Pode levar a falsas expetativas, isto é, o utilizador muitas vezes pensa que o *software* está terminado. Pacotes de *software* pobres devido ao objetivo principal do método, o desenvolvimento rápido. É impossível determinar com exatidão o tempo que o projeto vai demorar a ser desenvolvido. Não há forma de saber o número de iterações que serão necessárias.
Método em Espiral (anos 80)	As iterações iniciais do processo de desenvolvimento são menos dispendiosas, permitindo que as tarefas de maior risco sejam concebidas com menor custo. Componente análise de risco disponibiliza uma ferramenta de medida.	Aplicação complexa, implicando muitos anos de prática para aplicar o método com eficácia.
Métodos Ágeis (anos 90)	Os utilizadores (clientes) estão envolvidos ativamente durante o projeto.	Não são adequados para pacotes de *software* grandes, estáveis e com requisitos bem definidos. Os pedidos informais para melhorias, após cada fase podem gerar confusão.

Com as informações da Tabela 3, percebemos que a principal característica dos métodos ágeis é o envolvimento dos utilizadores no processo de construção do software. Além disso, os outros métodos possuem como desvantagens a inflexibilidade dos procedimentos, inviabilidade operacional, complexidade de implementação, longo tempo para que o produto fique pronto, entre outros fatores que nos levam a considerar que os métodos ágeis podem contribuir para o desenvolvimento de softwares educativos.

Diante do exposto, construído até aqui, acreditamos que os métodos ágeis podem contribuir para o desenvolvimento de SE na medida em que pretendemos envolver os utilizadores (pesquisadores, professores e estudantes) no projeto de desenvolvimento. Existem 12 princípios norteadores para a utilização das metodologias ágeis, que foram criados por um grupo de pesquisadores com a finalidade de orientar um novo paradigma de desenvolvimento de software, chamado de Manifesto Ágil, são eles:

1. Nossa maior prioridade é satisfazer o cliente desde o início por meio de entrega continua de software com valor agregado;

2. Modificações de requisitos são bem vindas, mesmo que no final do desenvolvimento. Os processos ágeis aproveitam as modificações como vantagens para competitividade do cliente.

3. Trabalho de entrega frequente de softwares funcionando, a cada duas semanas até dois meses, de preferência no menor espaço de tempo.

4. Executivos e desenvolvedores devem trabalhar juntos diariamente durante todo projeto.

5. Construção de projetos em torno de indivíduos motivados. Forneça-lhes o ambiente e apoio que precisam e confie que eles farão o trabalho.

6. O método mais eficiente e efetivo de levar informação para dentro de uma equipe de desenvolvimento é a conversa face a face.

7. Software funcionando é a melhor medida de progresso.

8. Processos ágeis promovem desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante, indefinidamente.

9. Atenção contínua a excelência técnica e ao bom projeto facilitam a agilidade.

10. Simplicidade – a arte de maximizar a quantidade de trabalho não efetuado – é essencial.

11. As melhores arquiteturas, requisitos e projetos surgem de equipes auto organizadas.

12. Em intervalos regulares, a equipe reflete sobre como se tornar mais efetiva, então sintoniza e ajusta adequadamente seu comportamento.

(MANIFESTO AGIL, 2001, tradução nossa).

Em síntese, encontramos nas metodologias ágeis subsídios suficientes para os princípios de funcionamento da equipe pluridisciplinar. Essa equipe, para a construção de um software de ensino e a aprendizagem de conhecimentos matemáticos, deve conter profissionais como Psicólogos, Pedagogos, Matemáticos, Engenheiros da Computação, entre outros, que possam trazer significativas contribuições para o desenvolvimento. As metodologias ágeis norteiam o trabalho da equipe, bem como as interações, o envolvimento, e como cada profissional pode contribuir com o desenvolvimento efetivo do software.

Ao considerarmos uma “equipe pluridisciplinar” acreditamos que na concepção e desenvolvimento de softwares educativos é importante reunir especialistas das diversas áreas que possam trazer contribuições para o melhor aproveitamento do software. Com isso, nossa proposta apresenta uma metodologia que viabiliza a sistematização de diferentes áreas do

conhecimento (necessárias para o desenvolvimento de softwares educativos para o ensino da Matemática) e organiza as interações entre esses diversos profissionais.

Com isso, justificamos a utilização de alguns dos princípios das metodologias ágeis a partir de duas reflexões importantes:

- como foi observado nas discussões até aqui construídas, para a concepção/desenvolvimento de softwares educativos, a constituição de uma equipe pluridisciplinar é de fundamental importância, entretanto, apenas a constituição da equipe não é suficiente, é necessário também estabelecer princípios metodológicos que organizem e favoreçam as interações, a formulação e reformulação das questões de pesquisa nos diversos domínios de conhecimentos envolvidos.

- a importância da confrontação das construções computacionais e das hipóteses que as justificam, no caso de SE, necessita da utilização de metodologias que integrem a observação da utilização do desenvolvimento por usuários finais, assim percebemos nas metodologias ágeis a possibilidade de integrar usuários e desenvolvedores no processo de engenharia do produto final.

3 ENGENHARIA DIDÁTICO-INFORMÁTICA

Das potencialidades e limitações dos atuais processos de desenvolvimento de software educativo observados na revisão de literatura realizada, percebemos a necessidade de articulação teórico-metodológica entre a Engenharia de Softwares e a Engenharia Didática. Neste capítulo discutimos a contribuição dessas engenharias para esse trabalho.

Iniciamos com algumas considerações sobre a Engenharia de Softwares, analisando as características aplicáveis para a construção de softwares educativos. Posteriormente, justificamos a escolha do trabalho com a engenharia de requisitos. Por último, apresentamos como a Engenharia Didática pode ser articulada com as engenharias supracitadas apresentando a Engenharia Didático-Informática.

3.1 Considerações sobre o desenvolvimento de softwares

Ao situarmos este estudo na problemática da engenharia de softwares faz-se necessário compreender o que é um processo de desenvolvimento e os modelos que existem para a construção. Com isso, concordamos com Sommerville (2004, p. 5) quando afirma que a engenharia de software "é uma disciplina da engenharia que se ocupa de todos os aspectos da produção de software, desde os estágios iniciais de especificação do sistema até a manutenção desse sistema depois que ele entrou em operação".

Com a definição acima, compreende-se a criação de um software observando não apenas o produto final que será elaborado, mas a atenção para os processos que antecedem o trabalho de criação, ou seja, um planejamento, como também a análise do produto final após o mesmo entrar em operação. De acordo com o IEEE (1993) *apud* Pressman (2006), engenharia de software é a "aplicação de uma abordagem sistemática, disciplinada e quantificável, para o desenvolvimento, operação e manutenção do software; isto é, a aplicação de engenharia ao software" (p. 17).

Ambas as definições supracitadas estão diretamente relacionadas com a ideia de procedimento, metodologia ou percurso para a construção do software, o que a literatura define como "processo de software", segundo Sommerville (2004),

> Um processo de software é um conjunto de atividades e resultados associados que geram um produto de software. Essas atividades são, em sua maioria, executadas por engenheiros de software. Há quatro atividades de processos fundamentais comuns a todos os processos de software. Essas atividades são: 1. *Especificação do software*. A funcionalidade do software e as restrições em sua operação devem ser definidas. 2. *Desenvolvimento do software*. O software deve ser produzido de modo que atenda a suas especificações. 3. *Validação do software*. O software tem de ser validado para garantir que ele faz o que o cliente deseja. 4. *Evolução do software.* O software deve evoluir para atender às necessidades mutáveis do cliente (p. 7).

Observamos a importância dos processos de software na medida em que o uso de metodologias e procedimentos pode facilitar o trabalho da equipe envolvida no projeto de construção do software, seja na análise das necessidades dos usuários, na observação do funcionamento, na manutenção, entre outros. Ao utilizar o termo "cliente" nos deparamos com a pluralidade de usuários que a concepção de softwares educativos possui. Sejam os professores, alunos, equipe pedagógica, entre outros, as necessidades desses usuários com os *softwares* educacionais os distingue de outros tipos de recursos, como afirma Santos (2009),

> Um processo de desenvolvimento de softwares educativos tem especificidades que o distinguem bastante de um procedimento de desenvolvimento de aplicativos comerciais, bancários ou domésticos. O engenheiro de softwares educativos não tem diante de si um sistema fechado no qual usuários e proprietários de sistemas e subsistemas interagem entre si através de procedimentos pré-estabelecidos, previsíveis e perfeitamente traduzíveis em operações automáticas e informatizáveis. Pelo contrário, um sistema educativo, por mais simples que seja, traduz e delimita conhecimentos em processo dinâmico de comunicação e percepção. Consequentemente, o engenheiro de softwares educativos deve lidar com um conjunto de aspectos subjetivos que caracterizam o fenômeno educativo. (SANTOS, 2009, p 18).

O objetivo dessa pesquisa é criar um Processo de Desenvolvimento de softwares educativos, ou seja, especificamos a finalidade da contribuição para o ensino e a aprendizagem de conhecimentos utilizando o termo "educativos". Assim, concordamos com Tchounikine (2004), quando o autor utiliza o termo *EIAH –Environnement Informatique pour l'Apprentissage Humain* (Ambiente Informático para Aprendizagem Humana) para designar ambientes informatizados que tenham sido projetados para promover a aprendizagem humana. De acordo com ele,

> O termo EIAH tomado em seu sentido mais amplo abrange uma variedade de trabalho e sistemas. Sua característica principal é a ligação de uma **intenção didática** e um ambiente informatizado. Esta intenção didática pode ser limitada à organização do sistema de formação em que se integram artefatos tecnológicos ou também se reflete no próprio artefato. (TCHOUNIKINE, 2004, p. 2, *tradução nossa, grifo nosso*).

Ao destacarmos o termo "intenção didática" acreditamos que todo software pode ser educativo, dada uma finalidade, um objetivo didático em seu uso. De acordo com Santos (2009):

> [...] o software educativo propriamente dito é aquele desenvolvido com as finalidades educativas explícitas demandando, para subsidiar sua produção, procedimentos específicos, relacionados a um conhecimento aprofundado dos processos cognitivos humanos, seja ele de natureza lúdica (um jogo educativo) ou de conteúdo escolar (um software para o ensino de Química, por exemplo), seja ele estático (em cd-rom) ou distribuído (para a Internet). (SANTOS, 2009).

Diante do exposto, concluímos que um software educativo configura-se como um *EIAH*, no sentido proposto por Tchounikine (2002) e também considera-se a definição de

Santos (2009). Com isso, faz-se necessário compreender qual o paradigma atual do desenvolvimento de *softwares* educativos e quais são os resultados das pesquisas nessa área.

Algumas dessas pesquisas (BELLEMAIN et al, 2014; BENITTI et al, 2005; SANTOS, 2009) apresentam fundamentos para a construção de *softwares* educacionais: jogos, aplicativos, ferramentas de colaboração *online*, simuladores, tutoriais, entre outros. Essas pesquisas apresentam possíveis caminhos teóricos e metodológicos para o desenvolvimento de softwares, com isso, observando as possibilidades das pesquisas analisadas, optamos por concordar com Tchounikine (2002), no momento em que o autor observa um paradigma entre a exploração das potencialidades tecnológicas em detrimento das especificidades de aprendizagem,

> Esta visão tecnocêntrica, com base nas possibilidades tecnológicas mais do que nas especificidades de aprendizagem, com máquinas, levou à construção de dispositivos genéricos (disponível a partir de promotores da tecnologia mais relevante por especialistas de *EIAH*), mas alguns sistemas efetivamente utilizados. O trabalho de investigação dirigida para sistemas cujo valor educativo é claramente reconhecido e que são efetivamente utilizados ou prestes a ser (nós não corremos o risco, propondo uma lista de sistemas que poderia ser exaustiva, mas citamos alguns trabalhos franceses, como o Aplusix, Cabri-géomètre ou Roboteach) não foram concebidos a partir da tecnologia, mas com uma reflexão combinando o estudo dos problemas de capacidades de aprendizagem e de ambientes informatizados. Por isso, estamos interessados em um processo de design de *EIAH* com significado profundamente multidisciplinar. (TCHOUNIKINE, p. 8, 2002, *tradução nossa*).

Com isso, no sentido operacional do processo de desenvolvimento discutido nesta pesquisa, pretendeu-se elaborar um percurso de concepção de SE observando as potencialidades advindas da tecnologia e as contribuições das pesquisas teóricas sobre o ensino-aprendizagem. Ao considerarmos os resultados de pesquisas e as orientações dos documentos oficiais quanto à utilização de recursos tecnológicos, compreendemos a utilidade dos mesmos para a aprendizagem de conhecimentos. Porém, as mesmas pesquisas indicam que ainda há muito que ser melhorado nos softwares disponíveis no mercado e um dos maiores fatores para esse problema é a engenharia a qual os softwares são criados.

3.2 Engenharias: de requisitos, Didática e de Softwares

Por considerar a importância da articulação entre pesquisas sobre ensino e aprendizagem de conhecimentos e exploração de potencialidades tecnológicas, procuramos associar conhecimentos da Engenharia de Software com as contribuições das pesquisas na área da Didática da Matemática. Com isso, em um primeiro momento percebemos que existem possibilidades definidas, mas não restritas, para a construção de softwares: os processos de software.

Existem diversos modelos de processo de software, de acordo com Sommerville (2004) um modelo é "uma descrição simplificada de um processo de software, que é apresentada a partir de uma perspectiva específica. [...] um modelo de processo de software é uma abstração do processo real que está ocorrendo" (p. 8). Ao analisar os processos existentes e o indicativo de pesquisas sobre o desenvolvimento de softwares educativos (BELLEMAIN et al, 2014; COSTA e COSTA, 2013, GALVIS-PANQUEVA, 1997) percebemos a necessidade da inserção dos conhecimentos sobre o ensino e a aprendizagem. Assim, nos próximos parágrafos apresentamos a articulação inicial entre conhecimentos da Educação Matemática e Engenharia de Softwares.

3.2.1 Engenharia de requisitos

Ao compreendermos a necessidade da especificidade da Engenharia de *Softwares* Educativos, acreditamos que a engenharia de requisitos possui elementos que contribuem na identificação das exigências dos usuários dos softwares educativos (professores, alunos e outros interessados). De acordo com Sommerville (2004), a especificação de software é uma atividade que,

> [...] destina-se a estabelecer quais funções são requeridas pelo sistema e as restrições sobre a operação e o desenvolvimento do sistema. Essa atividade, atualmente, é frequentemente chamada de engenharia de requisitos; ela é um estágio particularmente importante do processo de software, uma vez que erros nesse estágio inevitavelmente produzem problemas posteriores no projeto e na implementação do sistema. (SOMMERVILLE, 2004, p. 46).

Concordamos também com Nuseibeh e Easterbrook (2000) ao considerar que,

> A principal medida de sucesso de um sistema de software é o grau em que se encontra com o propósito para o qual foi concebido. Em termos gerais, a engenharia de requisitos para sistemas de softwares - ER é o processo de descobrir a finalidade, através da identificação das partes interessadas e suas necessidades, e documentá-las de uma forma passível de análise, comunicação e posterior implementação (NUSEIBEH & EASTERBROOK, 2000, p. 37, *tradução nossa*).

Com isso, acreditamos que o processo de engenharia de requisitos é de fundamental importância para a construção de softwares educacionais, observadas as exigências de desenvolvimento desse tipo de produto. A especificação dessas necessidades será contemplada no levantamento e análise dos requisitos. De acordo com Sommerville (2004),

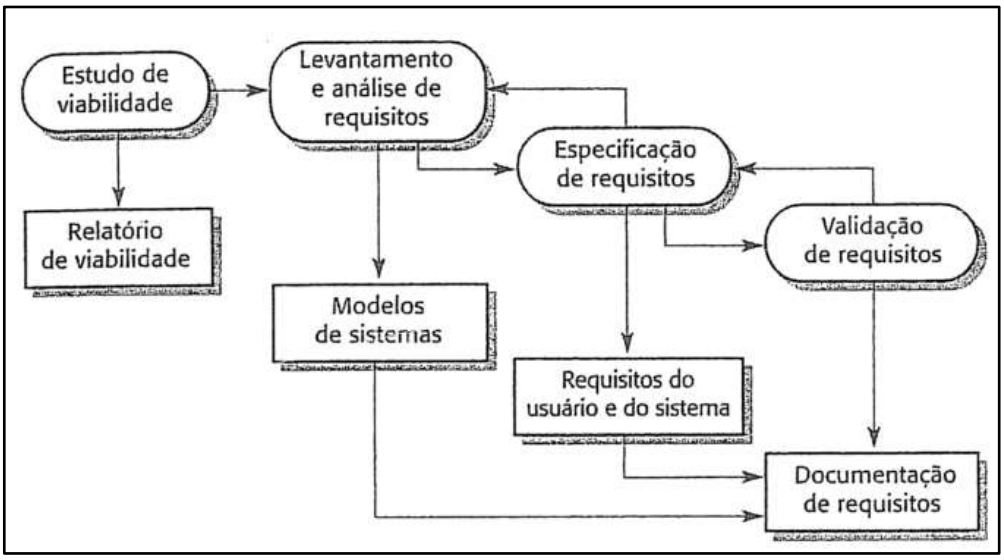
> Os problemas que os engenheiros de software têm para solucionar são, muitas vezes, imensamente complexos. Compreender a natureza dos problemas pode ser muito difícil, especialmente se o sistema for novo. Consequentemente, é difícil estabelecer com exatidão o que o sistema deve fazer. As descrições das funções e das restrições são os requisitos para os sistemas; e o processo de descobrir, analisar, documentar e verificar essas funções e restrições é chamado de *engenharia de requisitos* (SOMMERVILLE, 2004, p. 82, *grifo do autor*).

A engenharia de requisitos, de acordo com Zave (1997,*apud*NUSEIBEH & EASTERBROOK, 2000),

> é o ramo da engenharia de software que está preocupado com os objetivos do mundo real para as funções e restrições aplicáveis a sistemas de software. Está também preocupado com o relacionamento destes fatores para especificações precisas do comportamento do software e com sua evolução no tempo e através de famílias de produtos (NUSEIBEH & EASTERBROOK, 2000, p. 37, *tradução nossa*).

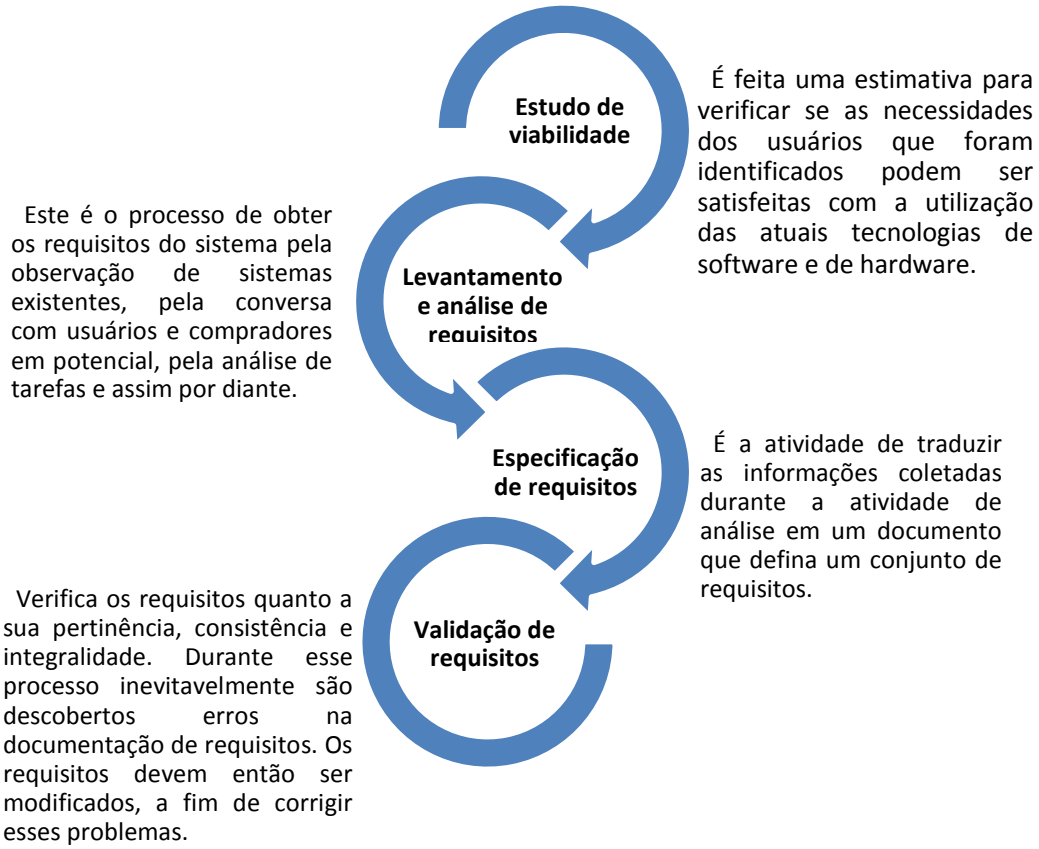
Compreendendo o cerne da engenharia de requisitos observamos a necessidade de articular os conhecimentos desse procedimento de software ao processo em desenvolvimento nesta pesquisa. Com isso, a utilização da engenharia de requisitos foi realizada sendo observadas as atividades propostas que, segundo Sommerville (2004), existe uma série de atividades genéricas comuns a todos os processos: elicitação; análise; validação e gerenciamento de requisitos. A Figura 4 apresenta o processo dessa engenharia segundo a autora.

Figura 4 – Processo de engenharia de requisitos (SOMMERVILLE, 2004, p. 47).



Porém, o que significa ***requisitos*** no momento em que tratamos de *softwares* educativos? A pergunta pode ser respondida sendo analisado o que se propõe em cada etapa da engenharia de requisitos, conforme a Figura 5.

Figura 5 – Atividades da engenharia de requisitos segundo Sommerville (2004)



Com a finalidade de evitar problemas durante o levantamento de requisitos, Sommerville (2004) sugere uma distinção entre diferentes tipos de requisitos: os *requisitos do usuário* – "são declarações, em linguagem natural e também em diagramas, sobre as funções que o sistema deve fornecer e as restrições sob as quais deve operar"; os *requisitos do sistema*, "estabelecem detalhadamente as funções e as restrições de sistema. O documento de requisitos de sistema, algumas vezes chamado de especificação funcional, deve ser preciso"; *Especificação de projeto de software* – "é uma descrição abstrata do projeto de software que é uma base para o projeto e a implementação mais detalhados". (SOMMERVILE, 2004, p. 82).

3.2.2 A Engenharia Didática e a pesquisa para a produção de softwares educativos

Ao considerarmos o atual cenário de desenvolvimento de software educativo para a concepção do modelo de processo de software desenvolvido nessa pesquisa, faz-se necessário observar as contribuições das pesquisas na área da Didática da Matemática. Sabendo que o processo de software visa o desenvolvimento de softwares educativos, consideramos de fundamental importância analisar o que é proposto atualmente para o progresso do ensino e da aprendizagem da Matemática. Assim, para aliar os pressupostos da Engenharia de Softwares com as pesquisas da Didática da Matemática, é possível utilizar o referencial teórico-metodológico da Engenharia Didática – ED (ARTIGUE, 1996). A noção de ED, segundo Artigue (1996),

> [...] emergiu em didática da matemática no início da década de 1980, com o objetivo de classificar uma forma do trabalho didático: aquela que era comparável ao trabalho do engenheiro que, para realizar um projeto preciso, se apoia nos conhecimentos científicos do seu domínio, aceita submeter-se a um controle de tipo científico, mas, ao mesmo tempo, se encontra obrigado a trabalhar sobre objetos muito mais complexos do que os objetos depurados da ciência e, portanto a estudar de uma forma prática, com todos os meios ao seu alcance, problemas de que a ciência não quer ou ainda não é capaz de se encarregar. (ARTIGUE, 1996, p. 193).

Ao analisar a ideia de trabalho didático, apresentada pela autora, acreditamos que o Engenheiro de Software Educativo realiza essa atividade ao se apoiar nos conhecimentos científicos do seu domínio utilizando uma metodologia para desenvolver seus produtos, mas trabalha com objetos complexos, pois como já observado nesse texto, a concepção de

softwares educativos é um processo que tem especificidades muito diferentes do desenvolvimento de aplicativos comerciais, bancários ou domésticos.

Nesses aplicativos também há uma complexidade para o desenvolvimento, mas ao nos depararmos com a necessidade das relações de ensino e aprendizagem, presentes nos SE, verificamos que atender tais situações é bastante difícil visto a pluralidade de usuários, as finalidades, os conhecimentos envolvidos, entre outros fatores. Com isso acreditamos que a utilização da ED articulada com a Engenharia de Softwares pode ser considerada de acordo com a perspectiva de Bellemain et al (2014),

> A Engenharia Didática (ARTIGUE, 1990, 2011), que trata da construção de sequências de ensino-aprendizagem a partir da utilização de conceitos e resultados de pesquisa, é objeto de reflexão de inúmeros estudos em didática da matemática. Nossa posição epistemológica é considerar que a concepção e o desenvolvimento de softwares educativos exige a mobilização de uma engenharia didática específica que deve integrar conceitos e métodos da informática. Esta engenharia também faz parte do domínio da engenharia de software, mas o desenvolvimento de um software educativo tem especificidades que o diferenciam de outros softwares (BELLEMAIN, et al, 2014, p. 6).

De acordo com Artigue (1996) a ED tem por finalidades analisar e propor situações didáticas, "A engenharia didática, vista como metodologia de investigação, caracteriza-se antes de mais por um esquema experimental baseado em *realizações didáticas* na sala de aula, isto é, na concepção, na realização, na observação e na análise de sequências de ensino". (ARTIGUE, 1996p. 196, *grifo do autor*). Nesse sentido, Artigue (1996) propõe a Engenharia Didática como metodologia de pesquisa conforme os procedimentos: análises prévias; construção e análise *a priori*; experimentação; análise *a posteriori* e validação.

Para melhor compreensão das fases e, principalmente, para que seja possível perceber a articulação com a Engenharia de Softwares organizamos abaixo algumas considerações importantes a serem feitas sobre cada etapa da Engenharia Didática, iniciando com a fase de análises prévias,

> Nas análises prévias é possível fazer uma análise aprofundada sobre o conhecimento em questão, de acordo com Artigue (1996) nessa fase é descrito um "quadro teórico didático geral em conhecimentos didáticos já adquiridos no domínio estudado, mas também apoiando-se em um certo número de análises preliminares que não, na maior parte dos casos: a ***análise epistemológica***; a ***análise do ensino*** e dos seus efeitos; a ***análise das concepções dos alunos***, das dificuldades e obstáculos que marcam a sua evolução; a análise do ***campo de circunscritores***; no qual virá a situar-se a

> realização didática efetiva; e naturalmente, tendo em conta os objetivos específicos da investigação. (ARTIGUE, 1996, p. 198, *tradução nossa, grifo nosso*).

Em específico no ‘campo de circunscritores’, a análise se efetua através da distinção de três dimensões: “epistemológica – associada às características do saber em questão, cognitiva – associada às características cognitiva do público ao qual se dirige o ensino, e didática – associada às características do funcionamento do sistema de ensino”. (ARTIGUE, 1996, p. 200).

Ao considerarmos as dimensões acima citadas, percebemos a ausência de uma dimensão específica, ou até exclusiva para tratar das questões tecnológicas. A partir disso, percebemos que seria possível desenvolver uma nova abordagem da Engenharia Didática em que a Dimensão Informática é inserida e trabalhada como as demais dimensões apresentadas. Bellemain et al (2015) nomeiam esse referencial teórico-metodológico de “Engenharia Didático-Informática”; utilizando a metodologia proposta na Engenharia Didática e na Engenharia de Software, articulada com uma análise de potencialidades tecnológicas aliadas a pesquisas teóricas.

Dando continuidade ao comparativo Engenharia Didática – Engenharia de Software, observa-se na fase de Concepção e Análise *a priori* o levantamento das variáveis de comando, essas variáveis direcionam ações a serem desenvolvidas na aplicação da sequência didática. São classificadas, segundo Artigue (1996), em dois tipos:

> - as variáveis macro-didáticas ou globais, que dizem respeito à organização global da engenharia;
>
> - e as variáveis micro-didáticas ou locais, que dizem respeito à organização local da engenharia, isto é, à organização de uma sessão ou de uma fase, podendo umas e outras ser, por sua vez, variáveis de ordem geral ou variáveis dependentes do conteúdo didático cujo ensino é visado (ARTIGUE, 1996, p. 202).

Tais variáveis, se comparadas ao desenvolvimento de softwares educativos, podem ser exemplificadas ao delimitarmos um campo do conhecimento matemático e pensarmos em soluções globais (macro) e soluções locais (micro). Como exemplo, poderíamos definir alternativas macro-didáticas para o desenvolvimento de um software que trabalhe o conceito de função polinomial do primeiro grau: Possibilidade de múltiplas representações e articulação entre elas (gráficos, tabelas, diagramas, linguagens algébricas e naturais); Conexão das funções com contextos diversos, etc. Definir essas alternativas é pensar como elas seriam

efetivadas em situações específicas do software: Quais situações viabilizariam as conexões entre as representações? Quais contextos devem ser sugeridos para ilustrar o conceito trabalhado? Etc.

Ao compararmos as variáveis de comando da ED com a Engenharia de Software, consideramos que as variáveis de comando podem fornecer os *requisitos iniciais* do software. Tais requisitos foram acessados através das pesquisas realizadas na área da Didática da Matemática, pois sabendo que o modelo de processo de software que será desenvolvido, tem por objetivo desenvolver produtos que atendam as necessidades específicas do ensino e da aprendizagem de conhecimentos matemáticos, faz-se necessário acessar as pesquisas nessa área. Além das pesquisas, documentos oficiais e avaliações nacionais de grande escala apresentam informações relevantes para essa temática.

Na fase de Concepção e Análise a *priori*, apresentam-se também as hipóteses de alternativas ao que fora levantado nas Análises Prévias, com isso a sequência é construída objetivando-se superar os problemas descritos na fase anterior. O processo de análise a *priori* serve para embasar a sequência de ensino, com uma base teórica fundamentada nas pesquisas na área, de acordo com as dificuldades expostas na análise prévia. Essa fase, em comparação com os métodos de desenvolvimentos de software, representa as etapas de construção do protótipo (SOMMERVILLE, 2004), nelas, os requisitos são agrupados e os testes de algumas unidades são feitas isoladamente, como descrito anteriormente.

A etapa da experimentação, na ED, consiste em testar a sequência didática que foi produzida. Seja para um grupo de estudantes da Educação Básica, Superior, entre outros, essa etapa consiste em por em prática o que foi construído. De igual modo ocorre nos processos de software, em determinado momento da concepção do produto faz-se necessário testar o que foi produzido até então. Na grande maioria dos processos de software, gera-se um protótipo e esse é testado verificando as falhas, os acertos, entre outras características.

Em relação à última fase, análise a *posteriori* e validação, têm-se a finalidade de confrontar a análise a *priori* e validar a sequência criada. Essa fase permite tratar os dados colhidos na experimentação verificando as contribuições da sequência aplicada, o que funcionou e o que não funcionou. Segundo Artigue (1996), a análise a *posteriori*,

> [...] se apoia no conjunto dos dados recolhidos durante a experimentação: observações realizadas nas sessões de ensino, mas também produções dos alunos na sala de aula ou fora dela [...] E, como já indicamos, é no confronto das duas análises, a *priori*, e a *posteriori*, que se fundamenta essencialmente

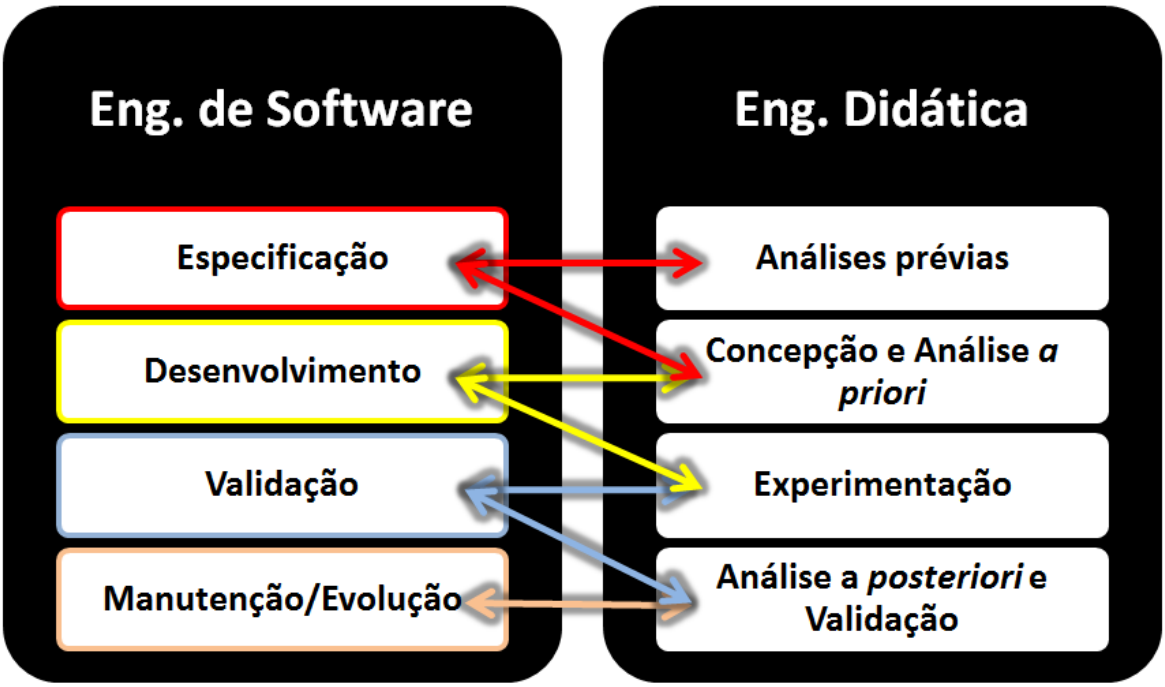
> a validação das hipóteses envolvidas na investigação (ARTIGUE, 1996, p. 208).

Ainda observando as semelhanças entre processos de software e a ED, percebe-se que a fase de "operação e manutenção" do software possui características comuns com a fase de análise a *posteriori* e validação, na medida em que na proposta de processo de desenvolvimento, consideramos que a análise a *posteriori*, revela elementos de grande importância para a evolução do produto.

Um dos elementos, talvez o principal, que diferencia as duas engenharias discutidas é o produto final de ambas. Como observado, a ES tem por produto final um software e a ED uma sequência didática (bem como a análise de sua aplicação). A ED fornece uma sequência para o aluno (e o professor como mediador), e a ES fornece um artefato (ou ambiente) que em geral ainda precisa de uma situação didática para chegar ao aluno (levando em consideração a natureza do artefato produzido).

Construímos no quadro abaixo as relações que são observadas entre a Engenharia Didática e de Software com o objetivo de melhor esclarecer as relações que estabelecemos entre as duas. Os processos não são equivalentes, mas as relações que se estabelecem entre eles nos fez perceber as possibilidades de articulação entre as duas engenharias.

Figura 6 – Comparativo entre as engenharias



Em resumo, os processos de desenvolvimento de softwares podem ser designados nas quatro etapas ilustradas na Figura 6: 1. Especificação, 2. Desenvolvimento, 3. Validação e, 4. Manutenção/Evolução. As etapas definidas na ED servem de fundamentação teórica e metodológica para o desenvolvimento de SE em nossa concepção de desenvolvimento de software educativo – Engenharia Didático-Informática – EDI .

A etapa de *Especificação*, momento em que se define a tipologia do software, requisitos e características, relaciona-se com as análises inicias da Engenharia Didática, observando que a metodologia de busca desse referencial delimita variáveis fundamentais para uma compreensão inicial dos requisitos do SE. A etapa de *Desenvolvimento* leva em consideração o levantamento teórico que foi realizado na Concepção e Análise *a priori* e nesse momento é iniciado o processo de experimentação do software.

O processo de *Validação* do software relaciona-se com a experimentação da ED e com a última fase, a análise a posteriori e validação. Validar o software significa verificar se o mesmo realiza o que se propõe a fazer. Fazem parte desta fase os testes, a criação de situações de utilização bem como a utilização do software nas situações de uso. Na ED a validação das sequências didáticas consiste em confrontar as hipóteses levantadas na análise a *priori* com a experimentação da sequência. Nesse sentido, consideramos que a validação de um software educativo pode ser realizada a partir de uma análise teórica: as situações de uso, funcionalidades e objetivos do SE que foram definidos na análise a *priori* são confrontados com o que se conseguiu desenvolver na experimentação e, com as conclusões desta investigação, o software é implementado ou o desenvolvimento é concluído.

Por último, a etapa de manutenção/evolução relaciona-se com a análise a *posteriori* e validação no momento em que a análise comparativa fornecida pela ED, traz a tona elementos que servem para o aperfeiçoamento e evolução do software desenvolvido.

Com isto, acreditamos que a utilização da Engenharia Didática traz elementos importantes para a concepção, desenvolvimento e análise de softwares educativos. Com a observação do caráter experimental dessa metodologia de pesquisa percebemos que o processo de desenvolvimento de um software é um experimento no sentido da ED. Compreendemos que a aproximação dessas duas engenharias (Didática e de software educativo) baseia-se na preocupação com as relações de ensino e aprendizagem de conteúdos

específicos: uma com a criação de sequências de ensino, a outra com a criação de produtos que auxiliem o ensino e a aprendizagem.

3.3 Engenharia Didático-Informática

O termo "Engenharia Didático-Informática" constitui-se na utilização dos procedimentos metodológicos de duas engenharias: Didática e de Softwares. Utilizamos tal expressão para designar esse estudo, pois o mesmo fundamenta uma engenharia de software com os contributos teóricos e metodológicos da Engenharia Didática. Assim, em estudos recentes, observamos que,

> A escolha da ED para auxiliar nosso trabalho de concepção e desenvolvimento tem também seus limites. Os elementos da ED como instrumento metodológico não são necessariamente todos pertinentes, como não são suficientes para resolver todas as questões relativas levantadas pela criação *software* educativos para matemática. Entretanto, ela auxiliou efetivamente nosso trabalho sistematizando questionamentos importantes para essa criação (BELLEMAIN et al, 2015, p. 6).

As limitações da Engenharia Didática são verificadas no momento em que tal metodologia não contempla, em suas contribuições, a totalidade das necessidades para o desenvolvimento de softwares educativos. Como fora observado, a ED não foi criada para auxiliar no desenvolvimento de softwares. Entretanto, ao analisarmos as fases da ED percebemos que a criação e o desenvolvimento de sequências didáticas têm características comuns com os processos de softwares existentes.

A partir dessa percepção e da verificação da necessidade, constatada em pesquisas, sobre o desenvolvimento de softwares educacionais, em articular as teorias sobre ensino e aprendizagem e os conhecimentos da Engenharia de Softwares, resolvemos associar os conhecimentos de tais engenharias, a de software, a didática e a de requisitos, criando a Engenharia Didático-Informática. Tal associação ocorre ao passo que se percebe a ausência de elementos para o desenvolvimento de produtos que contemplem necessidades do ensino e da aprendizagem da Matemática e das tecnologias (aspectos didáticos, cognitivos, epistemológicos, tecnológicos, entre outros).

A Engenharia de Softwares, por sua vez, não contempla especificidades que os softwares educativos necessitam. Diante disso, observamos a necessidade de reunir os elementos pertinentes das duas engenharias: A ED com os elementos de investigação teórica e experimental sobre o ensino e a aprendizagem e a ES com a padronização do desenvolvimento de softwares e métodos de obtenção de requisitos, com a finalidade de construir um processo de desenvolvimento de softwares educativos que observem os avanços tecnológicos, mas que não desprezem os estudos teóricos realizados para o ensino e a aprendizagem dos conhecimentos.

Com isso, propusemos uma nova abordagem de desenvolvimento de produtos tecnológicos para o ensino e a aprendizagem de conhecimentos matemáticos (e outros possíveis): A Engenharia Didático-Informática.

4 DESENVOLVIMENTO DO PROCESSO DE SOFTWARE

Neste capítulo encontra-se o direcionamento da pesquisa para alcançar os objetivos propostos. Partindo de uma revisão de literatura, de uma articulação teórica, da interação com uma equipe de orientação, estudo e desenvolvimento, foi elaborada a primeira versão do Processo de Desenvolvimento de Software, fundamentado na Engenharia Didático-Informática. Posteriormente, com base no levantamento inicial realizado deu-se início ao aperfeiçoamento e concepção da versão final do modelo de processo de software educativo discutido nesse estudo.

4.1 Percurso flexível e colaborativo de desenvolvimento

Ao considerarmos um percurso metodológico flexível, acreditamos na possibilidade de refazer alguns dos procedimentos pensados. O retorno à revisão de literatura, ao aporte teórico ou uma nova observação das sugestões e orientações de pesquisas publicadas, foram considerados em nossos procedimentos investigativos.

É válido ressaltar o caráter colaborativo do presente estudo, na medida em que obtivemos dados importantes em parceria com outros pesquisadores. Em específico, o estudo de Silva (2014) está diretamente associado aos resultados do presente estudo. Além de contribuir com o desenvolvimento do modelo de processo baseado na EDI, o pesquisador utilizou os resultados iniciais da presente pesquisa, nos proporcionando a possibilidade da realização de um estudo de caso: a partir das ideias iniciais do processo de desenvolvimento de SE discutido neste estudo, desenvolvemos um software para analisar e validar o processo de software criado.

Este caráter colaborativo de associação de objetivos de pesquisa possibilitou a construção e a análise, simultaneamente, com o referencial da Engenharia Didático-Informática, do desenvolvimento de um SE. Nas próximas sessões apresentamos como foi criado o processo de desenvolvimento de SE e como a interação entre a equipe de desenvolvimento ocorreu.

4.1.1 Composição da equipe pluridisciplinar e interações realizadas

A equipe pluridisciplinar foi constituída por membros do Grupo de Pesquisa LEMATEC. A tabela abaixo resume as funções e formações dos integrantes.

Tabela 4 – Equipe pluridisciplinar

NOME	FUNÇÃO	FORMAÇÃO
FRANCK BELLEMAIN	**ORIENTADOR DO ESTUDO/ENGENHEIRO PROGRAMADOR**	Graduado em Mathématiques, Mestre e Doutor em Didactique des Mathématiques
RICARDO TIBURCIO	**ENGENHEIRO-PESQUISADOR**	Licenciado em Matemática e Especialista em Ensino de Matemática
VERONICA GITIRANA	**ORIENTADORA DO ESTUDO**	Bacharel e Mestre em Matemática, Doutora e PHD em Educação Matemática
CESAR THIAGO SILVA	**PESQUISADOR-USUÁRIO**	Licenciado em Matemática
ANDERSON DOUGLAS	**MEMBROS COLABORADORES**	Licenciado em Matemática e Especialista em Ensino de Matemática
EDESON SIQUEIRA		Licenciado em Matemática e Mestre em Ensino das Ciências
EMANUELLA FRANÇA		Licenciada em Expressão Gráfica
JULIANA ARARIPE		Licenciada em Matemática e Mestra em Educação Matemática e Tecnológica
NUBIA SOUSA		Licenciada em Expressão Gráfica
ROBERTO MARIANO		Licenciado em Matemática e Mestre em Educação Matemática e Tecnológica
ROSILÂNGELA LUCENA		Licenciada em Ciências com Habilitação em Matemática. Especialista em Matemática, em Mídias na Educação e em Gestão Escolar. Mestre em Educação Matemática e Tecnológica
WILSON PEREIRA		Licenciado em Matemática e Especialista em Ensino de Matemática

Designamos por *Orientador do estudo/Engenheiro programador* o responsável pela orientação da presente pesquisa e pela programação do software na linguagem computacional. Por *Engenheiro-Pesquisador*, designamos o pesquisador responsável pelo direcionamento da presente pesquisa. Classificamos dessa forma por compreender que ao mesmo tempo em que este membro desenvolve a pesquisa também participa do processo de desenvolvimento do software.

A classificação de *Orientadora do estudo* faz referência à orientação da investigação do *Pesquisador-usuário*, pois como exposto, duas pesquisas estão articuladas e os orientadores das mesmas também. O termo *Pesquisador-usuário* indica que este integrante realiza sua pesquisa e também é agente do desenvolvimento do software. É classificado como usuário por propor situações de utilização do software para o ensino e a aprendizagem de conhecimentos matemáticos.

Por último, os *Membros-colaboradores* tem papel fundamental em todo o processo: tanto na criação do processo de desenvolvimento de SE, quanto no desenvolvimento. Estes

integrantes trazem para a discussão suas experiências, seus estudos, seus resultados de pesquisas, entre outros para auxiliar nesta investigação. Além disso, são considerados usuários finais e, observando os princípios dos métodos ágeis, a integração dos usuários no processo de desenvolvimento é interessante na medida em que situações de utilização e outras funcionalidades podem ser descritas por eles.

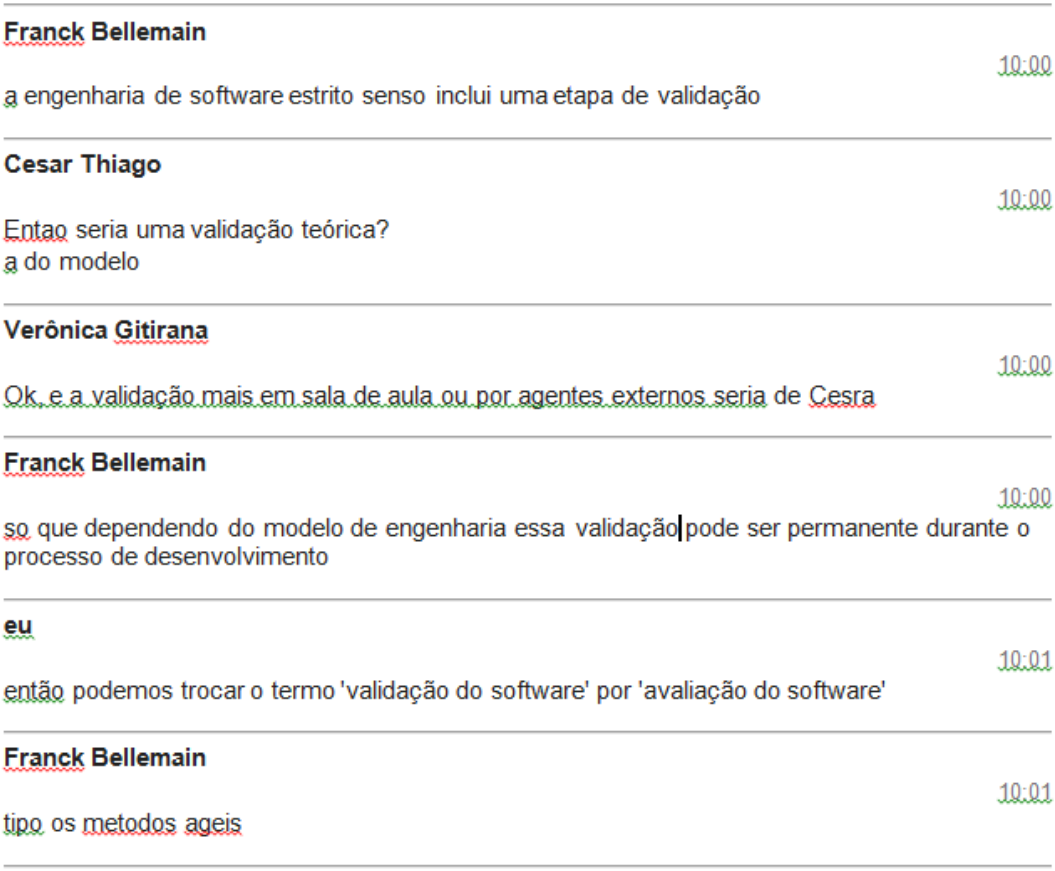
Com a equipe e as funções definidas iniciamos as interações necessárias para o desenvolvimento do processo de SE e do software. No total, contabilizamos seis encontros da equipe (completa); oito reuniões online apenas com os orientadores, pesquisador-usuário e engenheiro-pesquisador, três entrevistas online e diversas outras interações entre os membros (encontros de orientação, atividades desenvolvidas no Programa de Pós-Graduação em Educação Matemática e Tecnológica – EDUMATEC, e-mails, entre outras). Estas interações estão explicitadas e discutidas nas próximas sessões.

4.1.2 Evolução das versões do modelo de processo

Com o levantamento realizado na revisão de literatura, observando as lacunas e sugestões para o desenvolvimento de SE, confrontamos as pesquisas analisadas com o aporte teórico adotado e discutimos quais seriam as carências apresentadas pelos processos de desenvolvimento atuais que fazem os produtos de software não contemplar as necessidades dos conhecimentos em questão. A partir dos estudos teóricos iniciais, iniciamos as discussões com a equipe de desenvolvimento para fundamentar e dar início ao desenvolvimento do processo de software.

A primeira interação da equipe de desenvolvimento foi realizada através da ferramenta online de desenho do Google Drive, a Figura 7 apresenta alguns momentos importantes da interação.

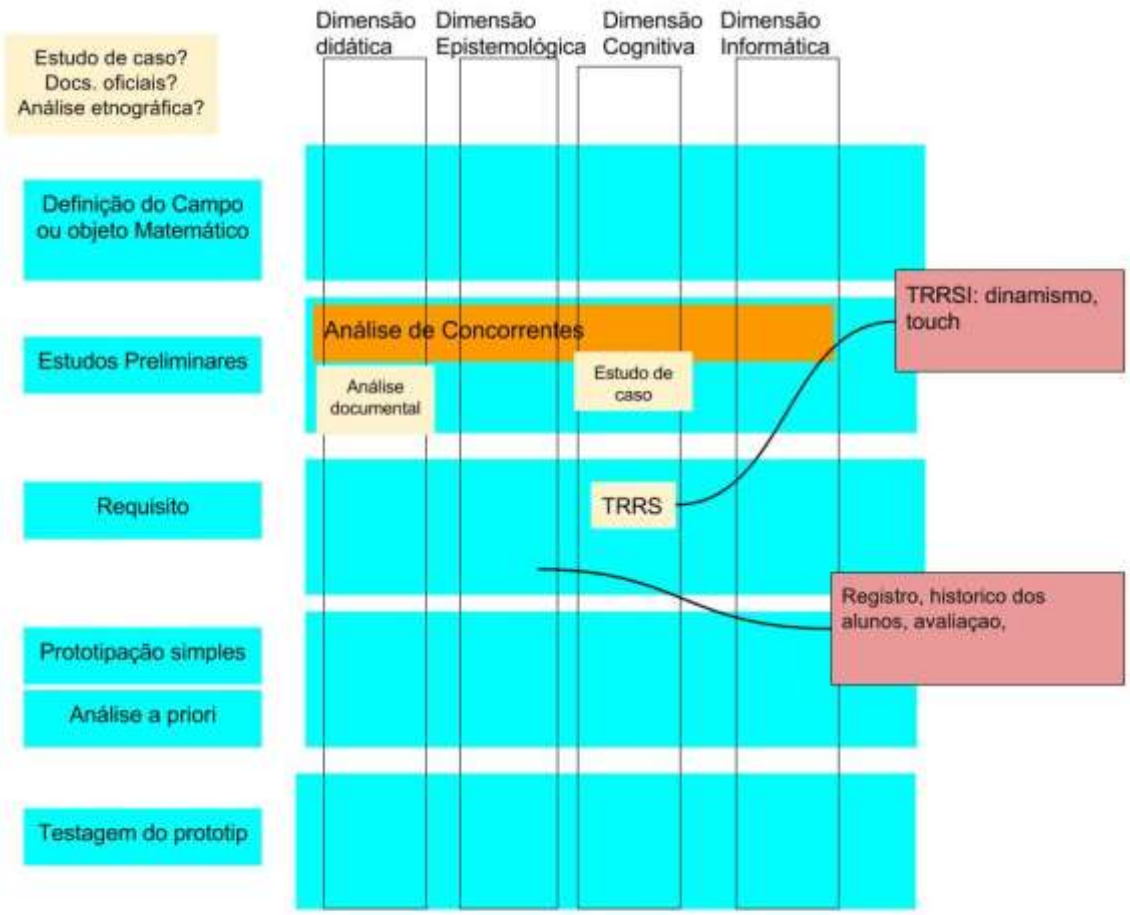
Figura 7 – Interação via chat.



Após algumas reuniões (online e presenciais), orientações e discussões sobre os processos de desenvolvimento de SE, compreendemos como fundamental considerar alguns elementos para a criação da primeira versão do modelo. Acreditamos que as dimensões (epistemológica, cognitiva, didática) indicadas no referencial da Engenharia Didática acrescidas da dimensão informática, deveriam estar articuladas com situações da Engenharia de Softwares. Além disso, observamos que o processo deveria ocorrer em etapas sequenciadas, mas caso fosse necessário, o processo poderia voltar a uma etapa anterior.

A Figura 8 apresenta a primeira versão do modelo de processo de software construído.

Figura 8 – Modelo inicial.



O modelo inicial foi construído por meio da análise da revisão de literatura já apresentada e o confronto com os pressupostos teóricos discutidos. Utilizando a ferramenta de desenho online do Google Drive (Figura 8), os orientadores do estudo, o engenheiro-pesquisador e o pesquisador-usuário, propuseram questionamentos, contribuições e observações sobre processos de desenvolvimento de SE que levassem em consideração as pesquisas da Educação Matemática. Embora a primeira versão careça de efeitos visuais sofisticados, tal produção foi de fundamental importância para o desenvolvimento da versão atual do modelo de software.

Os procedimentos observados na Figura 8 (definição do campo ou objeto matemático, Estudos preliminares, Requisito, Prototipação simples, Análise a priori e testagem do protótipo) relacionam-se com as dimensões assinaladas na horizontal (Didática, Epistemológica, Cognitiva e Informática) na medida em que questionamentos e observações

de tais procedimentos procurassem responder as referentes dimensões. A partir da versão inicial, reformulamos visualmente o modelo do processo para que a compreensão das etapas ficasse mais bem estruturadas e visíveis. A Figura 9 apresenta o modelo do processo com qual trabalhamos atualmente.

Figura 9 – Versão final do Modelo de Processo de SE

Além do aperfeiçoamento visual, dividimos os procedimentos em duas etapas: Fase teórica e Fase experimental. Essas fases não seguem uma sequência rígida, pois é possível que se revisitem alguns dos procedimentos iniciais para nortear o desenvolvimento do SE. É válido ressaltar, também, que a modelização exibe apenas as etapas norteadoras de desenvolvimento. Cada etapa ilustrada no modelo concerne diversos direcionamentos para a construção do SE. Tais direcionamentos também passaram por um processo de formulação e reformulação até a versão final do processo.

4.1.2 Descrição dos procedimentos

a) Delimitação do campo

Nesse procedimento inicial, define-se qual conhecimento pretende-se abordar no software que será desenvolvido. Ao utilizarmos a palavra "campo" acreditamos que o conhecimento não está isolado e, associado a esse conhecimento, outros conceitos podem surgir. Definem-se nessa etapa quais conhecimentos matemáticos serão abordados com o software, quais são os conhecimentos relacionados que também devem ser trabalhados, etc. e quais profissionais podem auxiliar nesse desenvolvimento. Nesta etapa, delimita-se também a equipe pluridisciplinar e as possíveis contribuições de cada profissional para a concepção e desenvolvimento do SE.

b) *Análises preliminares*

O procedimento de análises preliminares é o conhecido da Engenharia Didática, porém com a inserção da Dimensão Informática. Nesta etapa é realizado um levantamento direcionado a conhecer quais são os encaminhamentos Didáticos, Epistemológicos, Cognitivos e Tecnológicos do conhecimento delimitado. Essas análises contemplam os resultados das pesquisas em Educação Matemática sobre o campo de conhecimentos que será abordado. Realiza-se um apanhado teórico sobre o campo de conhecimentos para dar início ao processo de levantamento de requisitos. Ao ser concluída a análise, as variáveis de comando norteiam o desenvolvimento do SE fornecendo os primeiros requisitos.

c) Análise de requisitos

Após as análises preliminares, teremos o que é considerado na ED de "campo de circunscritores". Nesta análise, o que fora levantado no procedimento anterior é investigado e, com o apanhado teórico realizado, encontram-se os requisitos para a produção do SE. Utiliza-se aqui uma engenharia de requisitos baseada no modelo apresentado por Sommerville (2004), porém com um olhar para as dimensões do modelo de processo. Levantam-se nessa etapa os requisitos do sistema e os requisitos do SE a fim de atender as demandas apresentadas na revisão teórica realizada.

d) Prototipação e Análise a priori

Nesta fase, são pensadas nas situações de uso, nos problemas que podem surgir com a utilização, nas hipóteses de respostas dos usuários finais e no desenvolvimento do protótipo para iniciar os testes. Além disso, nessa fase verifica-se o funcionamento do protótipo para que eventuais erros sejam corrigidos antes da fase de testes.

e) Fase Experimental: Piloto, Professores e Alunos.

Este procedimento serve para testar o protótipo de software. Verificar falhas, sugestões de melhorias (interface, comandos, botões, etc) e verificar se atende aos objetivos dos usuários. Sugerimos a experimentação com um grupo controle, posteriormente com Professores (que também são usuários finais) e estudantes.

f) Análise a posteriori e Validação

Como na ED, a etapa da análise a *posteriori* consiste em confrontar as hipóteses com o que ocorreu na experimentação. Consideramos que essa análise é de fundamental importância para o aperfeiçoamento do SE. Confrontar o estudo teórico realizado com a experimentação traz elementos de análise que contribuem para o aprimoramento do produto.

O procedimento de Validação consiste na conclusão da análise realizada. Nesse momento, verifica-se se o conjunto (fase teórica e experimental) contribui para o ensino e a aprendizagem de conhecimentos matemáticos. Esta fase busca compreender se o software atende aos requisitos que foram levantados. Considerando as dimensões da EDI (Epistemológica, Cognitiva, Didática e Informática) e os requisitos nelas levantados, analisa-se a eficácia do software em atendimento dos requisitos que foram definidos. Essa verificação é realizada, também, a partir dos elementos que foram levantados na fase de experimentação do software. Os problemas identificados na utilização, as sugestões dos usuários e as demais solicitações realizadas, devem ser consideradas no processo de validação do SE em desenvolvimento.

4.1.3 Questionamentos para obtenção dos requisitos

Com a versão mais recente do modelo começamos a discutir cada fase nele apresentada. Elaboramos uma versão inicial de quais questionamentos deveriam ser feitos ao pesquisador-desenvolvedor com o intuito de obter os requisitos iniciais para o desenvolvimento do software e, simultaneamente, observar as alterações necessárias do modelo de processo em desenvolvimento. Os questionamentos iniciais foram disponibilizados online para o pesquisador-usuário via documento de editoração de apresentações do Google Drive, e a partir do momento que uma resposta era realizada, alguns questionamentos poderiam ser feitos online.

Ainda no âmbito do trabalho coletivo, em uma reunião presencial com a equipe pluridisciplinar, os questionamentos foram discutidos e, em relação ao procedimento de Delimitação do campo, chegamos aos questionamentos iniciais, apresentados na Tabela 5.

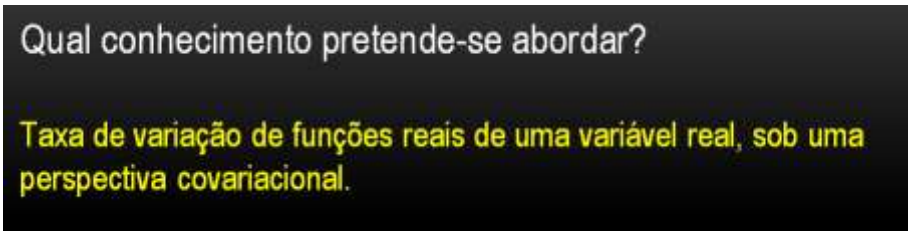
Tabela 5 – Questionamentos da Delimitação do campo

QUESTIONAMENTOS INICIAIS
Qual conhecimento pretende-se abordar?
Quais são as dificuldades de aprendizagem do conhecimento?
Por que as simulações configuram-se como opção para superar as dificuldades?
Quais as características fundamentais que o ambiente deve conter para atender as necessidades/características da aprendizagem do domínio?
Existem ambientes que trabalham com o domínio de interesse, quais são os recursos que contribuem para a aprendizagem e o que falta nesses softwares para contemplar as necessidades do domínio?

Ao expor os questionamentos iniciais para discussão do Grupo de Pesquisa LEMATEC, percebemos que havia a necessidade de torná-los mais direcionados, eliminando a possibilidade de respostas que não contribuíssem para o desenvolvimento do SE.

Analisando o primeiro questionamento: "Qual conhecimento pretende-se abordar?", observamos que as possíveis respostas poderiam ser diversas e não deixariam claro o suficiente para os desenvolvedores quais características possuem o conhecimento que se pretende trabalhar no SE. Utilizando o Google Drive, criamos um questionário online para que o pesquisador-usuário respondesse tais questionamentos. A Figura 10 indica a resposta do pesquisador-usuário para o questionamento proposto.

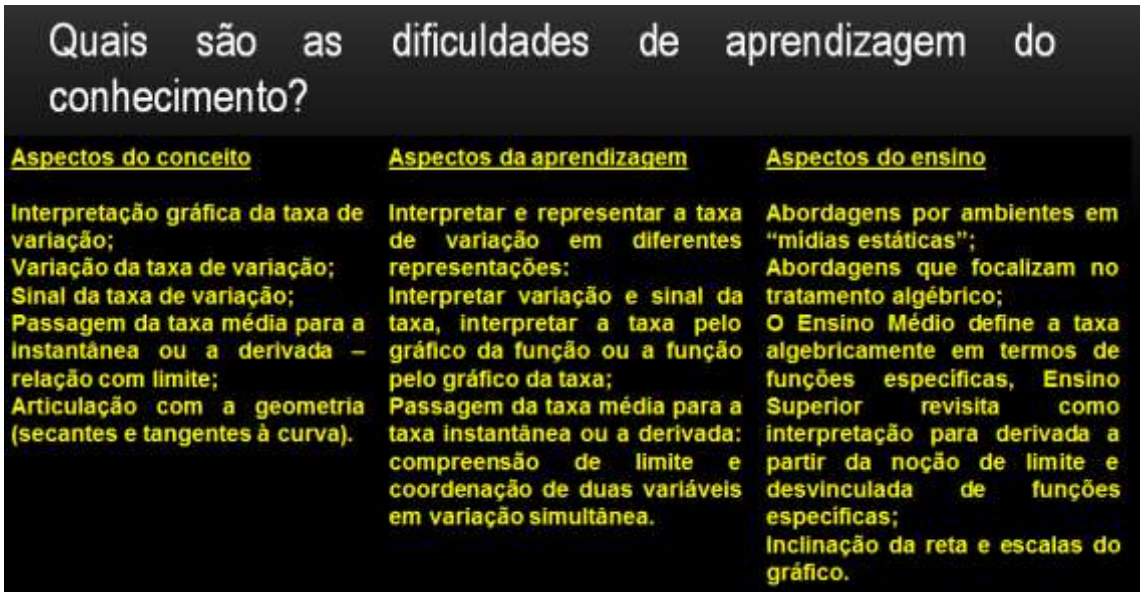
Figura 10 – Resposta do pesquisador-usuário – Conhecimento



Percebemos que a resposta apresentada não trazia elementos suficientes sobre qual abordagem e quais conhecimentos estariam associados. Além disso, não se compreendeu como seria a perspectiva citada, com isso, reformulamos o questionamento para melhor compreender as necessidades dos usuários finais.

Além disso, percebemos que o segundo questionamento deveria ser direcionado para outro momento, esse questionamento foi reformulado e foi posicionado para a parte das dimensões cognitivas e didáticas, entretanto, as respostas apresentadas foram fundamentais para que os requisitos do software fossem compreendidos, o que pode ser verificado na Figura 11.

Figura 11 – Resposta do pesquisador-usuário – Dificuldades de aprendizagem.

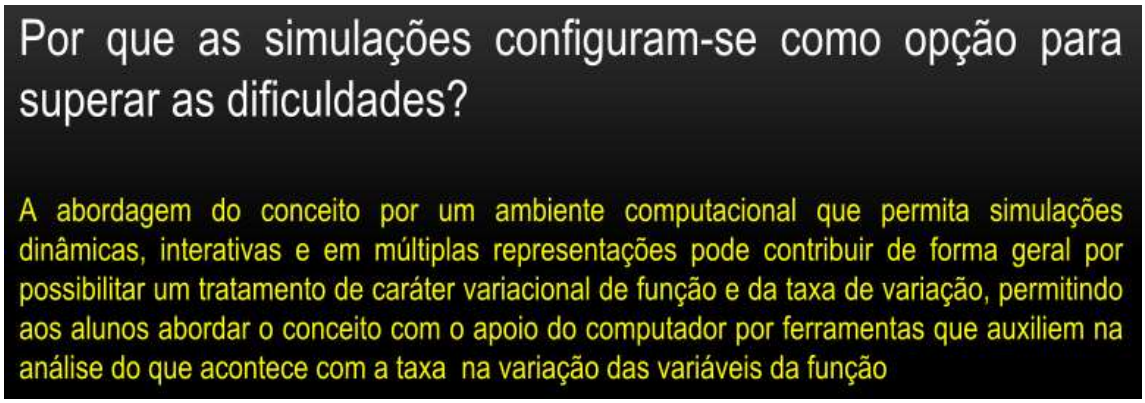


A partir da revisão da literatura e dos objetivos do pesquisador-usuário, compreendemos que o tipo de software que se adequaria a proposta idealizada seria um ambiente de simulações. Nas interações realizadas observamos que esse tipo de ambiente possibilita a superação das mídias estáticas viabilizando a compreensão dinâmica dos conceitos das funções matemáticas. Em uma das interações, viabilizada com a ferramenta de edição de textos do Google Drive, ao ser questionado sobre a tipologia do software, obtivemos o seguinte indicativo:

> • *Levando em consideração as tipologias de software educacionais existentes (tutoriais, jogos, simuladores, etc.) qual tipo de software pretende-se criar e quais são as justificativas para essa escolha?*
>
> *A abordagem do conceito por um ambiente computacional que permita simulações dinâmicas, interativas e em múltiplas representações pode contribuir de forma geral por possibilitar um tratamento de caráter variacional da função e da sua taxa de variação, permitindo aos alunos abordar o conceito com o apoio do computador por ferramentas que auxiliem na análise do que acontece com a taxa na variação conforme se varia as variáveis da função.*

Entretanto, para que o modelo de processo não ficasse restrito à construção de ambientes de simulação, reformulamos o questionamento para que outros tipos de software sejam desenvolvidos utilizando esse modelo. Ainda assim, considerando o questionamento proposto, obtivemos uma resposta, como pode ser observado na figura seguinte.

Figura 12 – Resposta do pesquisador-usuário: simulações.



Em relação aos dois últimos questionamentos, reunimos estes em um único e incluímos uma pesquisa sobre os softwares existentes, para que não haja a criação de um ambiente que já fora desenvolvido. Essa pesquisa é de fundamental importância, pois sabe-se que existe uma ampla variedade de SE que podem ser utilizados no ensino e aprendizagem da Matemática. Com isso, o usuário-desenvolvedor procura observar quais são os ambientes já desenvolvidos a fim de verificar as possibilidades de incrementos e inovação que podem ser desenvolvidas. As respostas aos dois últimos questionamentos podem ser verificadas nas figuras 13 e 14. A partir dessas respostas, conseguimos também identificar requisitos fundamentais para a elaboração do software.

Figura 13 – Resposta do pesquisador-usuário: características fundamentais.

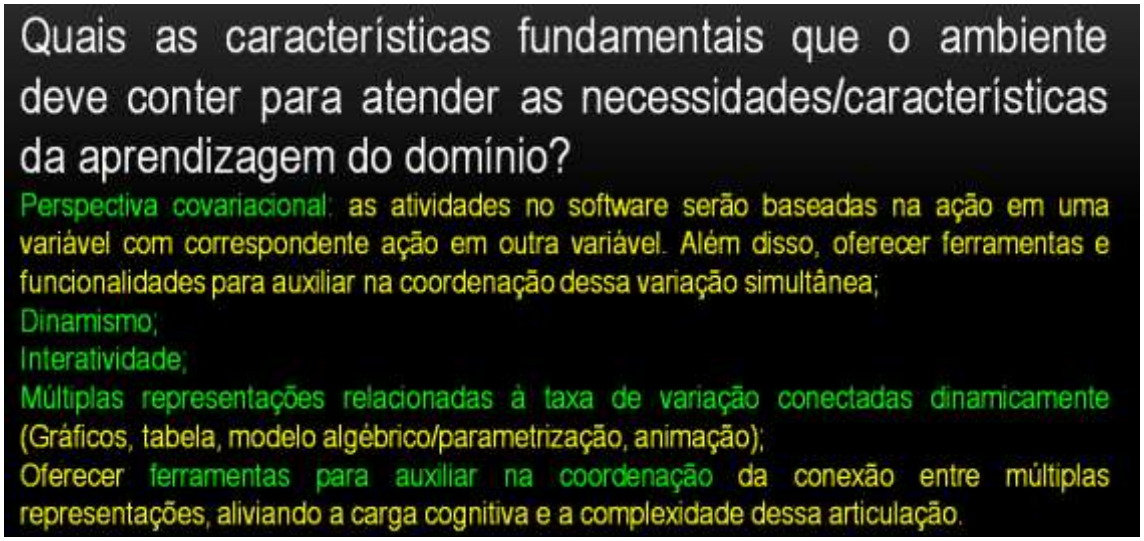
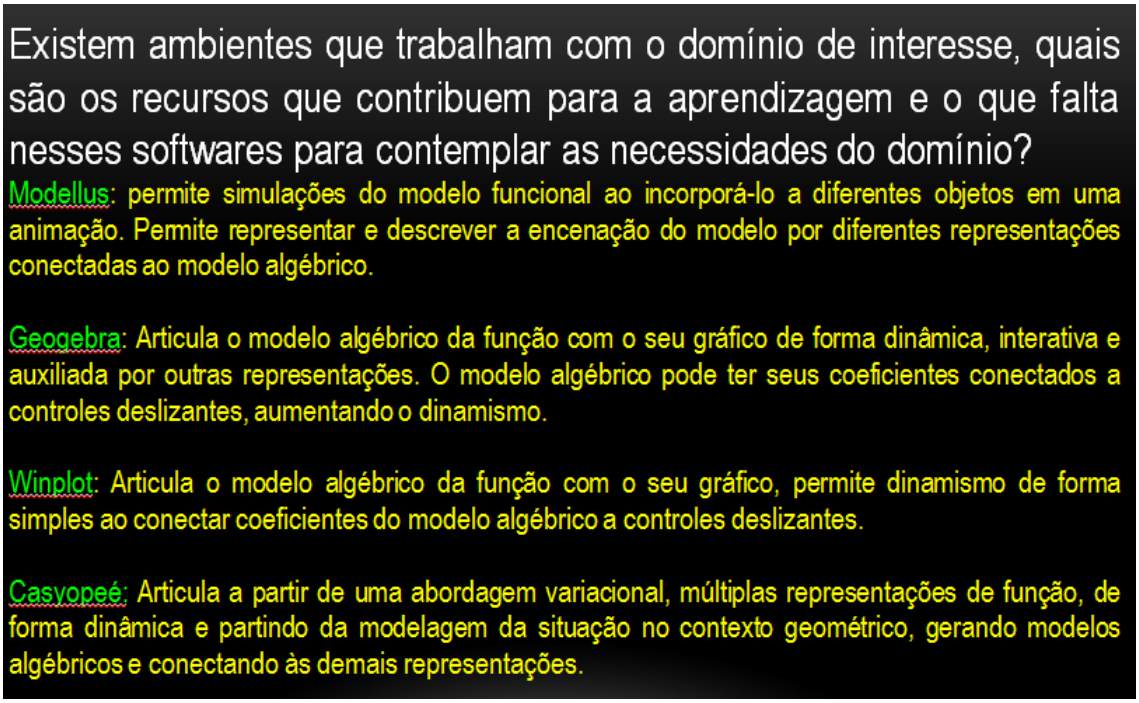


Figura 14 – Resposta do pesquisador-usuário: outros ambientes.



A partir das interações viabilizadas e das discussões promovidas, apresentamos, na tabela seguinte, uma análise comparativa dos questionamentos iniciais com os reformulados.

Tabela 6 – Questionamentos reformulados.

QUESTIONAMENTOS INICIAIS	QUESTIONAMENTOS REFORMULADOS
Qual conhecimento pretende-se abordar?	Qual campo de conhecimentos pretende-se abordar? Dentro deste campo de conhecimentos delimite conceitos e definições que serão trabalhadas e qual o foco que será dado ao conhecimento definido.
Quais são as dificuldades de aprendizagem do conhecimento?	Considerando as tipologias de software educacionais existentes (tutoriais, jogos, simuladores, etc.) qual tipo de software pretende-se criar e quais são as justificativas para essa escolha?
Por que as simulações configuram-se como opção para superar as dificuldades?	-
Quais as características fundamentais o ambiente deve conter para atender as necessidades/características da aprendizagem do domínio?	Existem ambientes que trabalham com o domínio de interesse, quais são os recursos que contribuem para a aprendizagem e o que falta nesses softwares para contemplar as necessidades do domínio?
Existem ambientes que trabalham com o domínio de interesse, quais são os recursos que contribuem para a aprendizagem e o que falta nesses softwares para contemplar as necessidades do domínio?	

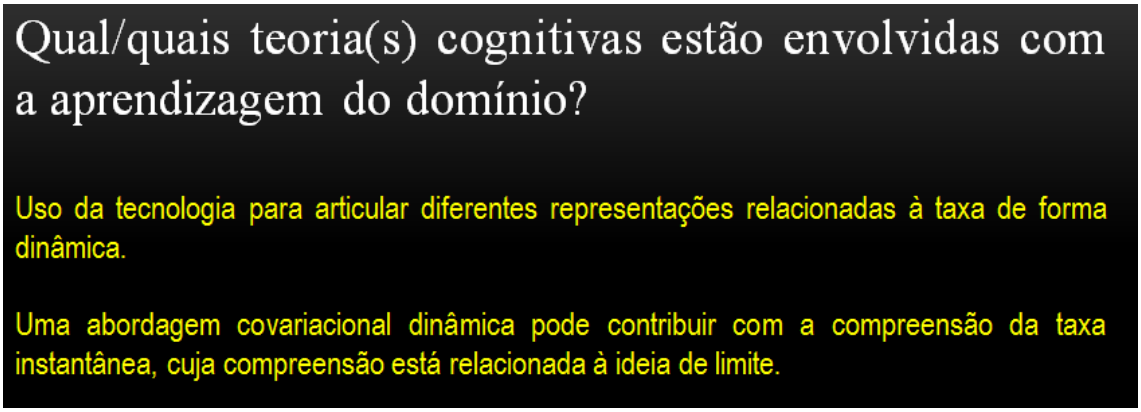
Em relação à formulação dos questionamentos da Análise preliminar, também foram realizadas modificações. Na Tabela 7 seguem os questionamentos iniciais e uma discussão sobre a reformulação dos mesmos em sequência.

Tabela 7 – Questionamentos das dimensões na Análise preliminar.

QUESTIONAMENTOS INICIAIS
DIMENSÃO COGNITIVA
1. Qual/quais teoria(s) cognitivas estão envolvidas com a aprendizagem do domínio?
DIMENSÃO DIDÁTICA
2. Qual é o estado atual do ensino do domínio? Quais são as contribuições e as principais dificuldades geradas pelo ensino atual?
DIMENSÃO EPISTEMOLÓGICA
3. Quais são as características do conhecimento que dificultam a aprendizagem e o ensino?
DIMENSÃO INFORMÁTICA
4. Como os recursos tecnológicos podem contribuir para o ensino e a aprendizagem do domínio?

Na reformulação do primeiro questionamento percebemos que uma resposta possível seria uma listagem de teorias associadas ao conhecimento proposto, o que não seria útil para o desenvolvimento do software. Com isso, reformulamos esse questionamento a fim de obter qual é o indicativo das teorias para o trabalho do conhecimento em questão, a Figura 15 exibe a resposta para esse primeiro questionamento.

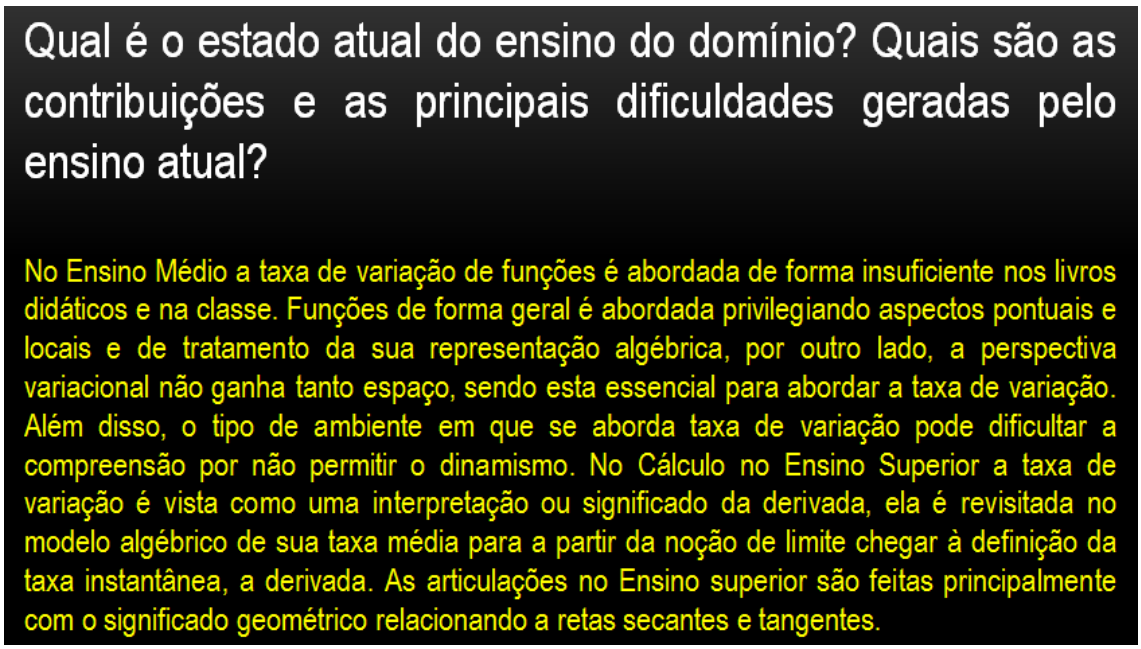
Figura 15 – Dimensão Cognitiva.



No questionamento exibido na Figura 15, não ficou claro o suficiente quais contribuições na área cognitiva o software deveria possibilitar. O pesquisador-usuário, em sua resposta, não adentra no caráter cognitivo, contudo ao analisar a resposta apresentada compreendemos que a situação cognitiva desejada estava relacionada com as possíveis representações dos objetos matemáticos bem como a articulação entre eles. Mesmo assim, realizamos outra interação com o pesquisador-usuário para refinar esse e outros questionamentos que não ficaram compreensíveis para o engenheiro-programador.

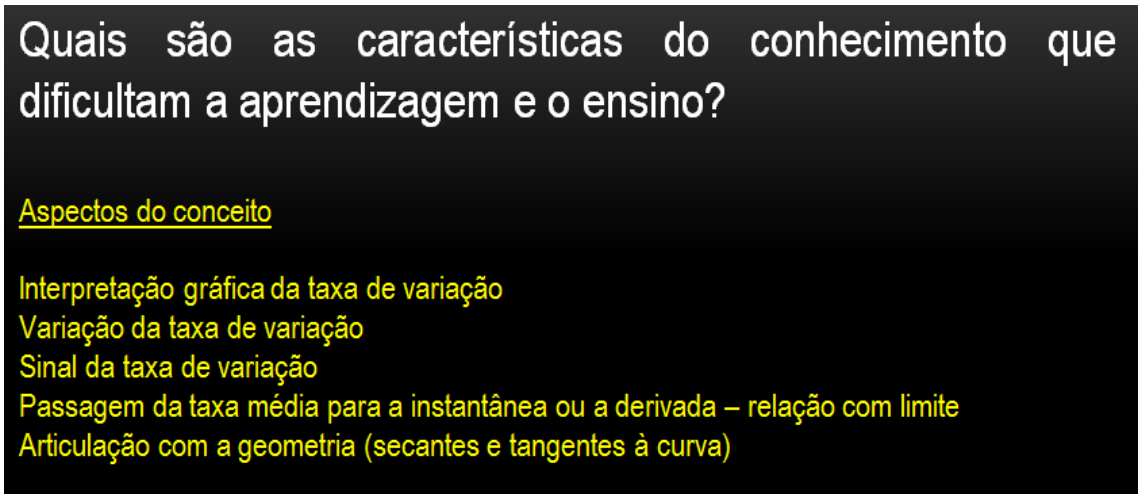
Na dimensão didática, a reformulação do questionamento se deu ao perceber que havia uma generalização de que o ensino atual gerava dificuldades para a aprendizagem dos diversos domínios de conhecimentos. Os elementos levantados na dimensão didática foram referentes à abordagem do conhecimento de funções no Ensino Médio e Superior. De acordo com o pesquisador-usuário, no Ensino Médio, a abordagem privilegia aspectos pontuais e locais enfatizando a representação algébrica. Verificou-se também que o tipo de ambiente em que se aborda taxa de variação pode dificultar a compreensão por não permitir o dinamismo, que pode ser viabilizado por ambientes de simulações. A figura seguinte apresenta a resposta do pesquisador-usuário.

Figura 16 – Dimensão Didática.



Na dimensão epistemológica a mudança foi sutil, retiramos a temática do ensino do questionamento e o tratando apenas das características do conhecimento, a figura seguinte exibe o questionamento antes da reformulação.

Figura 17 – Dimensão Epistemológica.



O último questionamento, referente à Dimensão Informática, foi reformulado com o intuito de conhecer quais características e como a tecnologia pode contribuir para o conhecimento que o software pretende trabalhar, o pesquisador-usuário já havia respondido esse questionamento quando o mesmo estava na Delimitação do Campo. Assim, apresentamos na Tabela 8 os questionamentos reformulados em comparação com os iniciais.

Tabela 8 – Reformulação da análise preliminar.

QUESTIONAMENTOS INICIAIS	QUESTIONAMENTOS REFORMULADOS
DIMENSÃO COGNITIVA	
1. Qual/quais teoria(s) cognitivas estão envolvidas com a aprendizagem do domínio?	1. Existem indicações na literatura de como o estudante aprende o conhecimento específico?
DIMENSÃO DIDÁTICA	
2. Qual é o estado atual do ensino do domínio? Quais são as contribuições e as principais dificuldades geradas pelo ensino atual?	2. Qual é o estado atual do ensino do domínio? Quais são as consequências desse ensino?
DIMENSÃO EPISTEMOLÓGICA	
3. Quais são as características do conhecimento que dificultam a aprendizagem e o ensino?	3. Quais são os aspectos do conhecimento que podem dificultam a aprendizagem?
DIMENSÃO INFORMÁTICA	
4. Como os recursos tecnológicos podem contribuir para o ensino e a aprendizagem do domínio?	4. Quais as características fundamentais que o ambiente deve conter para atender as necessidades/características que contribuam para o ensino e a aprendizagem do domínio?

Com os questionamentos reformulados e a primeira versão do processo de desenvolvimento de SE finalizado, deu-se início a construção do software educativo pretendido (no Apêndice A, encontram-se as respostas do pesquisador-usuário aos questionamentos reformulados). Criamos interações a fim de obter subsídios para a construção da versão final do software. Tais interações e o processo de desenvolvimento serão apresentados no capítulo seguinte.

5 PROCESSO DE DESENVOLVIMENTO DE SOFTWARE: UM ESTUDO DE CASO

Neste capítulo apresentamos os resultados da utilização do Processo de desenvolvimento de software educativo no cenário do Grupo de Pesquisa LEMATEC. Com o objetivo de desenvolver um produto para trabalhar com o conceito de Taxa de Variação de Funções Polinomiais, desenvolvemos um aplicativo para testar o modelo de processo aqui discutido.

5.1 Estudo de caso: Desenvolvimento de software para o ensino e a aprendizagem do aspecto variacional de funções

A partir das primeiras interações (reuniões com a equipe de desenvolvimento, respostas dos questionamentos, discussões sobre o software, etc.) conseguimos definir requisitos teóricos e funcionais necessários para o desenvolvimento do SE pretendido. Entretanto, mesmo com as interações realizadas, alguns termos utilizados nas respostas apresentadas pelo pesquisador-usuário não foram compreendidos para que fosse possível obter as características idealizadas nas análises preliminares. Com isso, utilizando a ferramenta de edição de textos do Google Drive, foi aberto um documento para refinar alguns dos questionamentos feitos, a fim de obter respostas mais precisas e delimitar os requisitos necessários para o desenvolvimento do SE.

A primeira incompreensão foi com o termo "simulações dinâmicas", não foi possível compreender a necessidade indicada com a resposta do pesquisador-usuário, com isso criamos mais uma interação (questionário online) e obtivemos a resposta conforme os diálogos em seguida.

> • *O que se entende por "simulações dinâmicas"? Quais recursos tecnológicos poderiam viabilizar tal situação desejada?*
>
> *A expressão simulações dinâmicas é usada no sentido de que os objetos do ambiente possam ser manipulados dinamicamente, como as variáveis em um gráfico de coordenadas ou um objeto ao qual é atrelado um modelo matemático que modele sua movimentação na tela. O termo também faz referência ao dinamismo na conexão entre notações e representações, por exemplo, ao variar um ponto no gráfico mostrar o valor da variável sendo modificado simultaneamente, ou ainda, mostrar uma sequência de valores em uma tabela que são modificados simultaneamente à variação no gráfico.*
>
> ***Sugestão do desenvolvedor**: Para a proposta de dinamismo no ambiente que será criado é possível utilizar o recurso touchscreen (bem como aproximação e distanciamento da tela possibilitados pelo recurso).*
>
> ***Réplica do pesquisador-usuário**: O Touchscreen potencializa ainda mais o dinamismo do ambiente por permitir um contato suave e contínuo com a tela, dando mais controle da ação de variar pelo sujeito e evitando movimentos pausados e particionados pelo mouse ou mousepad.*

Outra incompreensão foi referente ao conceito de interatividade, pois sabemos que esse termo é carregado de significados nas diversas áreas, por isso o motivo da dúvida, observe a resposta apresentada.

- *Qual o conceito de "interatividade" que se pretende obter no software?*

O sentido dado por Kaput (1992) para mídia interativa, no contexto de sua discussão, é o que vê a interação como uma contribuição física do sistema de notação e do meio no qual ele está instanciado, ou seja uma resposta do sistema a uma ação do usuário. Ele diferencia meio inerte e meio interativo, o primeiro é caracterizado como aquele em que a única mudança de estado, dada uma ação do usuário, é a exibição da entrada. O autor coloca como característica chave para as mídias interativas, a adição de algo novo a ação do usuário, requerendo sua resposta, ele aponta dois aspectos geralmente presentes em mídias interativas: "limitações ou suportes embutidos" (como por exemplo, a auto escala que define automaticamente as escalas do eixo y conforme o eixo x) e "agentes que realizam ações para o usuário virtualmente" (como realizar e checar cálculos, dar feedback da ação do usuário ou gravar ações e resultados para uso posterior).

Com a resposta apresentada conseguimos compreender qual o sentido da interatividade desejada. Em termos gerais, o pesquisador-usuário percebe a necessidade de respostas do software de acordo com a ação do usuário final. Além do aspecto de interatividade desejado, a questão das representações foi observada. O pesquisador-usuário desejava que o software possibilitasse múltiplas representações, além do que foi sugerido pelo engenheiro pesquisador, é possível observar o interesse do pesquisador-usuário diante deste aspecto, como pode ser observado no diálogo abaixo.

- *As múltiplas representações fazem referência aos possíveis registros de representação do conceito de função (algébrica, tabular, gráfica, etc)? Como se pretende viabilizar a visualização múltipla desses registros?*

***Sugestão do engenheiro-pesquisador**: A divisão da tela em partes, ou janelas que possam ser minimizadas (com o recurso de selecionar múltiplas, ou apenas uma) é uma opção para a representação de vários objetos ao mesmo tempo.*

***Réplica do pesquisador-usuário**: Além da possibilidade da divisão da tela e de selecionar uma ou mais representações para possibilitar o trabalho com as principais representações, é pretendido contemplar um dos aspectos levantados por Kaput (1992), a sobreposição de notações/representações, quando esta for possível. A sobreposição de notações visa possibilitar por exemplo sobrepor dados numéricos ao gráfico ou objetos da geometria dinâmica como uma reta secante para representar a taxa média de variação entre dois pontos. Além disso, pretende-se trabalhar com gráficos sobrepostos como o gráfico da função e da taxa de variação da função sendo traçados*

sobrepostos e simultaneamente ou um gráfico de barras dinâmico que se sobreponha ao gráfico para representar a variação da taxa média em um intervalo.

Por último, observamos a utilização do termo "suporte ao raciocínio dos alunos" e solicitamos melhores esclarecimentos sobre o referido termo, a resposta pode ser observada no diálogo seguinte.

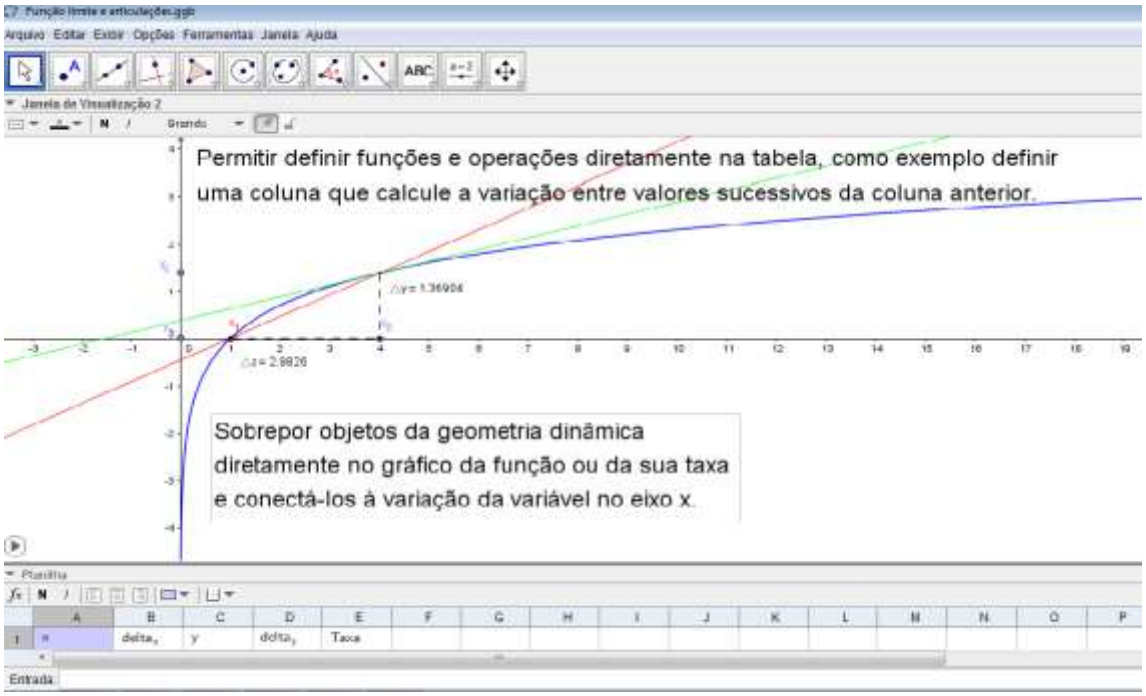
- *Como o se pretende configurar o "suporte ao raciocínio dos alunos"?*

Kaput (1992) refere-se a esse suporte como necessário principalmente em uma abordagem que conecte ações em diferentes sistemas de notação. Para o autor o processo de translações entre sistemas de notações demanda uma carga cognitiva acentuada, principalmente em um sistema onde as ações realizadas se perdem no decorrer da atividade do aluno, não permitindo o seu resgate. Exemplos de suporte ao raciocínio segundo o autor seriam a "gravação repetível de uma ação" para possibilitar a reprodução da ação e a desvinculação de uma sequência nas ações de translação entre sistemas, o que permite que ações em um sistema A possam ser refletidas em B imediatamente ou quando o sujeito estiver pronto para visualizá-las.
No contexto da taxa de variação e da perspectiva covariacional, ainda foram elencadas outras funcionalidades importantes para suportar o raciocínio dos alunos:

Rastro da variável (discreto e contínuo)
Variação automática (por um botão play) ou manual (ao manipular as variáveis e objetos ligados a ela)
Macros e possibilidade de definir novas operações (como definir uma nova função/operação em uma planilha)
Ferramenta de memorização da variação ou da taxa de um intervalo ao selecionar dois pontos do gráfico

Com as interações realizadas iniciamos o trabalho de levantamento de requisitos, porém, o pesquisador-usuário, com o intuito de expressar melhor algumas situações que o software deveria possibilitar, apresentou telas, construídas com o software Geogebra, para exemplificar situações desejadas no ambiente a ser desenvolvido. A Figura 18 apresenta a situação classificada como "Função, limite e articulações".

Figura 18 – Construção no Geogebra (SILVA, a publicar)



Com as construções no Geogebra, algumas das especificações do software pretendido foram percebidas. Na figura 18, o pesquisador-usuário indica que há a necessidade de utilizar mais de uma representação do objeto matemático em questão, além da articulação dinâmica entre essas representações. Além disso, observamos também o indicativo para a articulação do conhecimento levando em consideração diferentes áreas da Matemática (Álgebra, Geometria, Cálculo, entre outras).

Com a construção ilustrada na Figura 18 e as outras realizadas, o pesquisador-usuário exibe algumas das funcionalidades que o software deve possuir para contemplar os indicativos do seu levantamento teórico. Compreendemos o desenvolvimento dessas situações como uma necessidade do pesquisador-usuário de explicitar suas ideias e objetivos, o que foi considerado útil para o desenvolvimento do software pelo engenheiro-programador. As demais construções utilizando o Geogebra são apresentadas no Apêndice B.

Reunindo os elementos levantados nas análises preliminares e as interações com a equipe pluridisciplinar, realizamos um mapeamento dos requisitos do SE a ser desenvolvido. Na próxima sessão apresentamos tais elementos.

5.2 Mapeamento dos requisitos

Organizamos os requisitos os classificando nas dimensões propostas no modelo de processo aqui discutido, os requisitos foram elencados utilizando as respostas do pesquisador-usuário e a compreensão dessas nas interações que foram descritas nas sessões anteriores. A seguir o resultado do levantamento realizado.

Requisitos referentes à Dimensão Cognitiva

a) Possibilidade de articular e conectar diferentes representações relacionadas à taxa de variação, de forma dinâmica, o que supõe o uso de um ambiente que propicie a dinamização dessas representações.

b) Viabilização de perspectiva variacional: no contexto da covariação e de forma dinâmica, que pode contribuir com a compreensão da taxa instantânea.

Requisitos referentes à Dimensão Didática

a) Abordagem do conceito em "mídias estáticas": requer do aluno um esforço mental muito grande para visualizar a variação, um aspecto dinâmico.

b) Abordagens que focalizam o tratamento algébrico: limita uma visão variacional e fortalece uma visão pontual da função.

c) No Ensino Médio se define a taxa algebricamente em termos de funções específicas, já no Ensino Superior revisita como uma interpretação da derivada a partir da noção de limite e desvinculada de funções específicas: as abordagens têm focos distintos, no Ensino Superior o conceito de limite é fundamental.

d) Problemas na articulação com o contexto geométrico: aspectos como a articulação com a inclinação da reta tangente podem gerar dificuldades quando o conceito da tangente é simplesmente importado da geometria plana e não contextualizado no Cálculo e a inclinação dessa reta sofre mudanças visuais com as mudanças das escalas do gráfico, o que também traz problemas com a associação da noção de coeficiente angular da reta e a taxa de variação.

Requisitos referentes à Dimensão Epistemológica

a) Interpretação gráfica da taxa de variação: necessidade de fazer uma leitura variacional do gráfico e relacionar concavidade e pontos de inflexão com a taxa de variação.

b) Variação da taxa: a taxa de variação determina a mudança da função, mas a própria taxa pode ser constante ou variável, quando ela varia o aluno precisa lidar com a modificação desta taxa.

c) Sinal da taxa de variação: a taxa pode ser positiva ou negativa, foi relatada dificuldade dos alunos em interpretar a taxa quando negativa.

d) Passagem da taxa de variação média para a instantânea ou a derivada – relação com limite: o conceito de limite torna-se fundamental na compreensão da passagem da taxa média para a instantânea, logo é preciso conhecer esse conceito para entender tal passagem.

e) Articulação com a geometria: a passagem da taxa média para a derivada é fortemente articulada com a geometria no sentido de uma "aproximação" da reta tangente ao ponto por retas secantes a dois pontos, essa articulação traz em si questões como a importação do conceito de reta tangente da Geometria Plana que deve ser adaptado ao Cálculo. Além disso, as associações entre a derivada e o coeficiente angular da reta tangente, bem como os aspectos visuais do gráfico como a inclinação da reta e as mudanças de escalas do gráfico também merecem atenção na articulação com a geometria.

Requisitos referentes à dimensão Informática

a) Perspectiva covariacional

b) Dinamismo

c) Interatividade

d) Conexão dinâmica de múltiplas representações relacionadas à taxa de variação

e) Suporte ao raciocínio dos alunos por meio de ferramentas inseridas nos contextos acima.

A Tabela 9 resume os requisitos identificados nas interações com o pesquisador-usuário.

Tabela 9 – Quadro resumo dos requisitos.

COGNITIVOS	DIDÁTICOS	EPISTEMOLÓGICOS	INFORMÁTICOS
Múltiplas representações	Abordagem dinâmica (criação de situações)	Possibilidade de facilitar a interpretação gráfica	Simulações
Articulação entre as representações	Articulação com o contexto geométrico	Perspectiva variacional	Interação
Manipulação e conexão entre as representações	Abordagem variacional	Visualização da covariação	Dinamismo
		Relações com os conceitos do Cálculo	Suporte ao raciocínio dos usuários

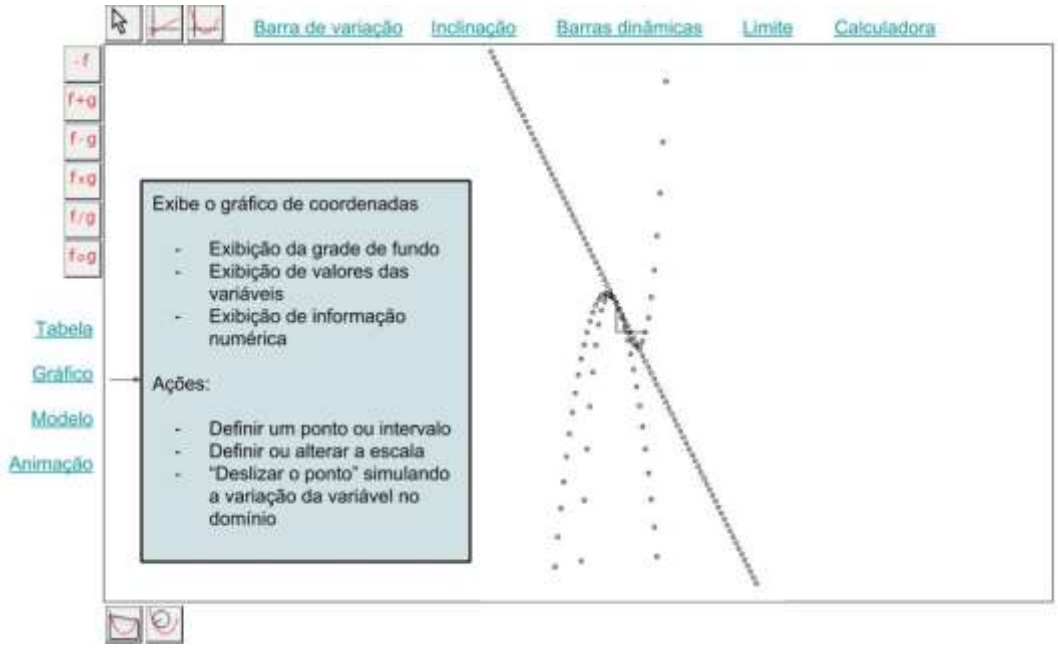
5.3 Protótipo do software

Com os requisitos definidos, iniciamos o processo de desenvolvimento do software. Utilizamos o método de prototipação de telas para compreender menus, botões, funcionalidades, recursos, que o software deveria conter. Esse método de prototipação consiste em idealizar a interface apresentando um esboço do funcionamento do software. Utilizando o Power Point, construímos em equipe as telas exibindo as funcionalidades idealizadas a partir dos requisitos levantados. Apresentamos a seguir algumas dessas telas, as demais se encontram no Apêndice B.

Figura 19 – Tela inicial do software.

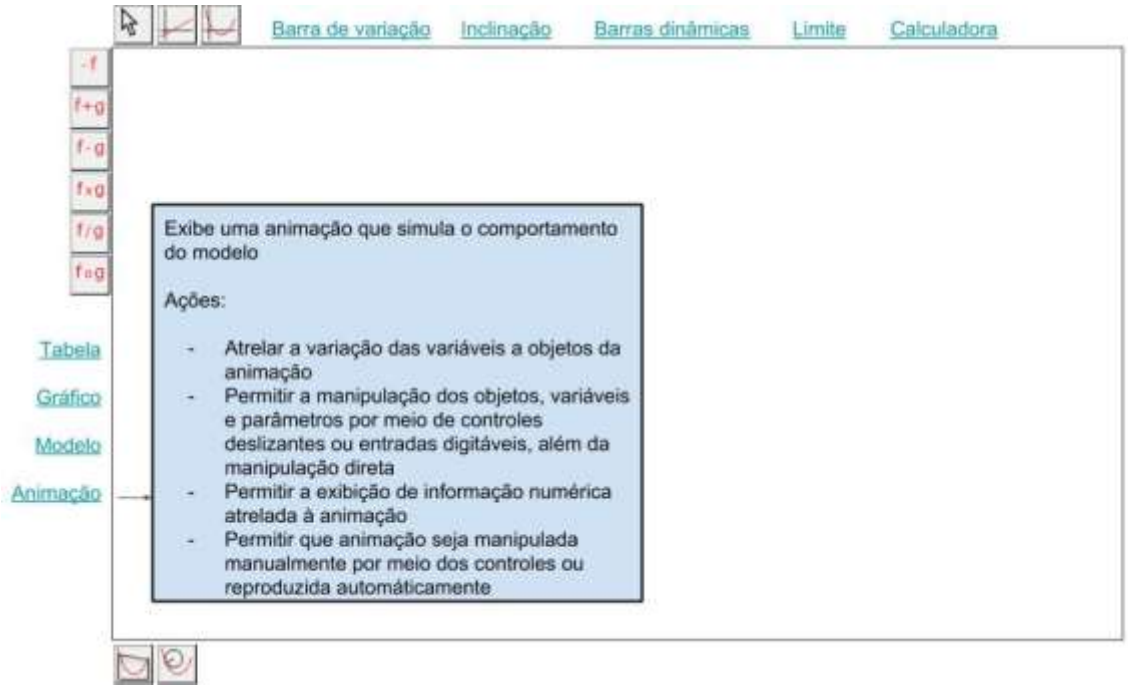
Na Figura 19 apresentamos a tela inicial do protótipo. Como já observado, o processo de prototipagem precede o desenvolvimento do software. Os botões e menus, nessa fase, ainda não estavam com seus layouts definidos, mas as funcionalidades pretendidas com eles, já estavam idealizadas. Na Figura 20, observamos um dos botões e sua funcionalidade descrita.

Figura 20 – Funcionalidades do menu "Gráfico".



A construção do protótipo reuniu elementos importantes para o desenvolvimento da primeira versão do software. A partir da reunião dos requisitos com as funcionalidades pretendidas foi possível construir o que fora idealizado. Por exemplo, uma das funcionalidades solicitada foi ilustrada com o menu “Animação”, como pode ser verificado na Figura 21. Com as ideias do pesquisador-usuário expostas no protótipo o desenvolvedor pode facilmente analisar e, a partir das limitações dos recursos tecnológicos, fazer com o que foi idealizado torne-se realizado no software.

Figura 21 – Botão Animação.



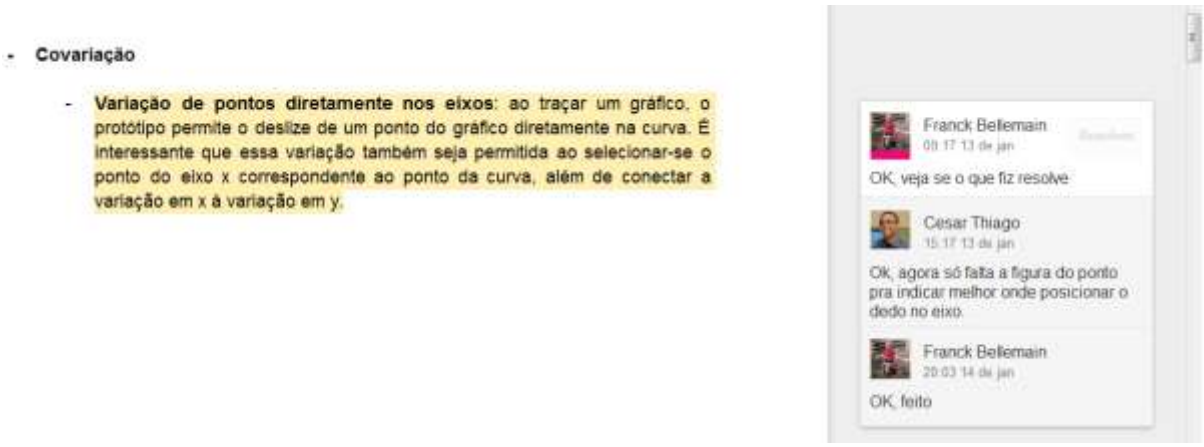
Em resumo, a fase de Análise a Priori e Prototipação, proposta no Processo de Desenvolvimento discutido nesse estudo foi útil para o processo de compreensão dos requisitos para o desenvolvimento do software educativo. Essa fase possibilitou a visualização das possíveis situações de uso, as funcionalidades, botões, menus, recursos, entre outros. Após o protótipo ter sido concebido e analisado, deu-se início ao desenvolvimento, bem como a programação, da primeira versão. As características da programação do software estão descritas no Apêndice D.

5.4 Retorno do pesquisador-usuário a primeira versão do software

Após o desenvolvimento da primeira versão do software o disponibilizamos para a análise da equipe. O software ficou disponível online e após a utilização do mesmo nos reunimos (pesquisador-usuário, orientadora do estudo, engenheiro-pesquisador e engenheiro-programador) para verificar quais ajustes deveriam ser feitos, quais funcionalidades poderiam estar em falta, etc.

Após uma reunião presencial decidimos abrir um documento online no Google Drive para que o cliente registrasse suas impressões e solicitasse tais ajustes. Alguns trechos dessa interação podem ser verificados nas figuras que seguem.

Figura 22 – Aspecto variacional.



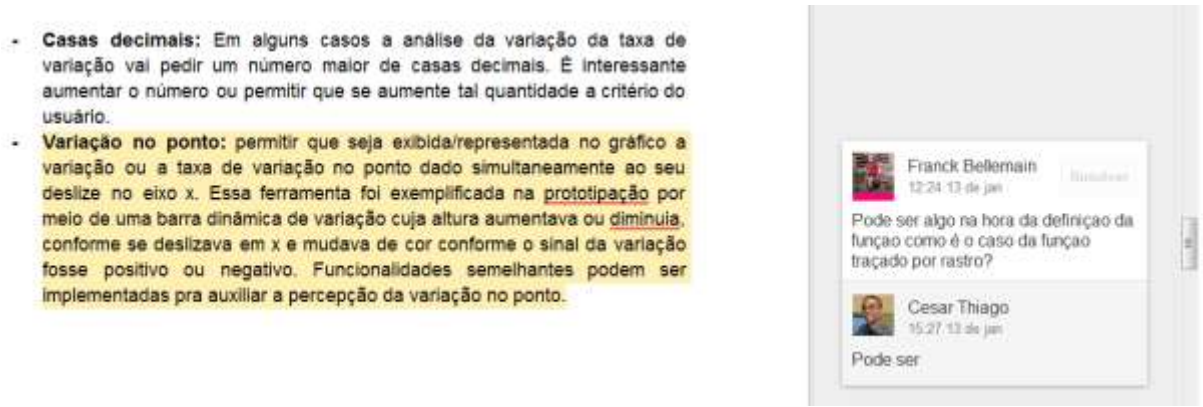
Na Figura 22 o cliente faz considerações a respeito de uma das funcionalidades do software e descreve o que seria interessante que o mesmo pudesse realizar. Dois aspectos são solicitados: a variação de pontos nos eixos e o rastro na construção do gráfico. No diálogo apresentado, o desenvolvedor mostra que fez as alterações no software e o pesquisador-usuário responde que ainda falta o melhoramento de outra funcionalidade.

Na Figura 23 continua-se a discussão sobre o aspecto variacional, porém agora com a funcionalidade do rastro. No diálogo o engenheiro-programador apresenta sua proposta para a solicitação, indicando que para versões futuras do software a funcionalidade solicitada será implementada. Há uma concordância com a alteração feita, mas ainda solicita-se que outras modificações sejam realizadas para que a fase de experimentação seja bem sucedida.

Figura 23 – Rastro para a construção do gráfico

Na figura 24 apresentamos outras funcionalidades que foram solicitadas. No primeiro item, solicita-se que o número de casas decimais seja aumentado para que o conhecimento que será trabalhado no software seja contemplado nos mínimos detalhes. Quanto ao segundo item, variação no ponto, o pesquisador-usuário indica que tal ferramenta foi exemplificada no protótipo e que a mesma é de suma importância para a usabilidade do software. Percebe-se ainda a justificativa da necessidade de tal ferramenta com a viabilização da percepção da variação no ponto, funcionalidade discutida nas análises dos requisitos.

Figura 24 – Outras solicitações.



Com a interação entre o engenheiro-programador e o pesquisador-usuário foi possível verificar quais modificações deveriam ser realizadas na primeira versão do software e as implementações do software foram realizadas. Consideramos de grande importância tal interação devido ao fato de alguns dos requisitos levantados não terem sido contemplados e, com o diálogo online, as características e funcionalidades esperadas foram inseridas. Além das colocações observadas nas figuras apresentadas, outros pontos foram discutidos e estão apresentados no Apêndice C. No próximo capítulo discutimos os resultados dessa pesquisa e como o desenvolvimento do processo de desenvolvimento de software educativo foi finalizado.

Com isso, concluímos a primeira versão do software e o mesmo encontra-se disponível para uso no seguinte link: http://lematec.net.br/funcao/#. Que ficará público após as implementações necessárias.

5.5 Análise *a posteriori* e Validação

Ao se considerar um processo de desenvolvimento de SE que articule teorias sobre ensino e aprendizagem de conhecimentos matemáticos acreditamos que uma das formas de validação desse processo é verificar se o produto criado atende as necessidades apontadas nas pesquisas teóricas. Com isso, analisamos os requisitos levantados em cada uma das dimensões apresentadas no modelo a fim de verificar as possibilidades de atendimento do software ao que se idealiza nos estudos teóricos.

Referente a Dimensão Cognitiva, o foco percebido na revisão de literatura realizada pelo pesquisador-usuário indica que a multiplicidade de representações seja viabilizada com os recursos tecnológicos. Os requisitos cognitivos levantados foram: múltiplas representações dos objetos matemáticos; articulação entre as representações e manipulação e conexão entre as representações. Todos esses requisitos podem ser verificados no software desenvolvido. A aprendizagem da taxa de variação de funções matemáticas, campo de conhecimentos abordados no software, pode ser viabilizada com a utilização de representações tabulares, gráficos cartesianos e linguagem algébrica entre outras.

Na fase de testes percebeu-se que o software possibilitava a visualização, articulação e manipulação de algumas dessas representações. A única representação que não está totalmente articulada com as demais é a tabular, visto que não há a possibilidade de modificar

um elemento da tabela e essa alteração ser visualizada em outras representações, a tabela apenas exibe as interações realizadas nas demais representações.

Quanto a Dimensão Didática, os indicativos teóricos fazem referência as possibilidades de abordagem dos conhecimentos. O pesquisador-usuário apresentou algumas situações desejadas nesta dimensão: possibilidade de criação de situações de ensino; articulação com o contexto geométrico e abordagem variacional. O software, por se tratar de um ambiente de simulações, permite que haja a construção de situações de ensino, como já fora observado, esse tipo de ambiente necessita estar inserido em uma situação didática para seu aproveitamento ser efetivado.

A articulação com conhecimentos da Geometria também foi considerada, construindo uma situação didática para a utilização do software, o mesmo pode contribuir para a compreensão dos conceitos de inclinação da reta, coeficiente angular e taxa de variação, solicitados pelo pesquisador-usuário. Requisitos referentes a Dimensão Epistemológica também foram contemplados com a possibilidade de articular outros conhecimentos. Referente a articulação com o Cálculo: composição e operações de funções, noção de limite, variação da variável, entre outros, podem ser trabalhadas com o software.

Os requisitos obtidos na Dimensão Informática, referiam-se a funcionalidades que o software deveria possibilitar: simulações, no sentido da manipulação de um objeto articulado com os demais; interação, o sistema deve prover respostas às ações do usuário. Uma situação que não contemplou-se totalmente foi o “suporte ao raciocínio dos usuários”. Com esse termo o pesquisador-usuário referia-se a possibilidade da gravação de algumas interações dos usuários finais no software. Nesse aspecto o rastro da variável foi contemplado parcialmente: existe a possibilidade de traçar a função através do rastro, mas nem sempre que a variável se desloca essa situação é possível.

Outro requisito não contemplado na Dimensão Informática foi a possibilidade de animação por parte do software: a variação automática está associada à animação e esse aspecto não foi implementado, o software não possui animação em nenhum dos tipos de função que consegue representar.

A validação que idealizamos para o software desenvolvido pode ser classificada como semi-teórica. Ao se considerar o referencial da Engenharia Didático-Informática, confrontamos as análises preliminares, bem como o levantamento teórico realizado sobre o ensino e a aprendizagem do conhecimento específico, com as possibilidades oferecidas pelo

software e com os requisitos que foram elencados. Com a análise a posteriori, mesmo observando que alguns requisitos não foram contemplados nesta versão, conseguimos validar o produto desenvolvido compreendendo que as implementações que não foram verificadas neste momento devem ser realizadas posteriormente.

O processo de validação do software será enriquecido com a experimentação do mesmo nas situações de uso que serão desenvolvidas pelo pesquisador-usuário. Os resultados desta experimentação serão de suma importância para a implementação do software e verificação de elementos que não foram contemplados com o desenvolvimento. Além disso, destacamos outras situações que não foram vislumbradas neste momento e as elencamos no próximo capítulo.

6 CONSIDERAÇÕES SOBRE A INVESTIGAÇÃO

Apresentamos neste capítulo a análise e a discussão dos resultados obtidos desde a concepção do modelo até a sua aplicação no estudo de caso. O processo de formulação e reformulação do software proposto nos forneceu informações precisas para o aperfeiçoamento do modelo e outros encaminhamentos que serão apresentados.

A partir dos objetivos desta investigação descrevemos, neste capítulo, a análise dos resultados obtidos bem como encaminhamentos futuros. Com isso, os tópicos em seguida abordam a perspectiva coletiva dos resultados; as observações sobre o referencial teórico metodológico desenvolvido, e encaminhamentos sobre o aperfeiçoamento do processo de desenvolvimento de software educativo criado.

6.1 Sobre a imersão do estudo no Grupo de Pesquisa LEMATEC

Uma das primeiras particularidades da realização desta investigação foi a perspectiva de associação de duas pesquisas com produtos finais distintos. Enquanto este estudo tinha por objetivo conceber e analisar em estudo de caso um processo de desenvolvimento de software educativo, outro pesquisador, denominado *pesquisador-usuário* observava a necessidade de artefatos tecnológicos para o ensino e a aprendizagem do conceito de taxa de variação de funções. A partir dessa situação, aliada a visão do Grupo de Pesquisas LEMATEC, observamos que a realização de pesquisas acadêmicas não está associada a apenas um pesquisador, mas a perspectiva de colaboração deve ser considerada.

Diante disso, com as convergências entre os objetivos das duas pesquisas, conseguimos obter resultados positivos com a articulação realizada. As interações (reuniões online e presenciais, formulários, entrevistas, questionamentos, e-mails, etc.) foram realizadas e trouxerem elementos de análise importantes para o estudo articulado.

Um primeiro resultado da interação realizada foi o levantamento bibliográfico sobre a concepção e desenvolvimento de softwares educativos. Ao idealizarmos uma metodologia de desenvolvimento que considerasse estudos teóricos sobre o ensino e a aprendizagem com tecnologias, fez-se necessário realizar uma ampla revisão de literatura para compreender a perspectiva atual de desenvolvimento. Nossa primeira hipótese, de que uma engenharia que alie os estudos teóricos sobre a aprendizagem e ensino com as potencialidades tecnológicas promove a criação de *softwares* educativos que atendam as necessidades específicas dos conhecimentos envolvidos, foi comprovada nos estudos analisados. Existe atualmente um direcionamento para o desenvolvimento de softwares educativos considerando estudos teóricos sobre o ensino e a aprendizagem dos conhecimentos. Entretanto, os produtos criados, em sua grande maioria, ainda estão divididos em duas classes: os que exploram as potencialidades tecnológicas ou os que estão fundamentados apenas nos estudos teóricos.

Outro resultado a ser considerado foi o levantamento bibliográfico realizado pelo pesquisador-usuário, a partir dos primeiros indicativos de como o software idealizado seria desenvolvido o pesquisador-usuário seguiu as orientações da primeira versão do modelo e realizou as *Análises Prévias* (conforme descrito no modelo criado) do campo de conhecimentos que o software iria trabalhar. A partir deste levantamento foi possível obter os primeiros requisitos e um direcionamento inicial para o desenvolvimento do software.

Por último, na perspectiva de grupo, destacamos os recursos tecnológicos utilizados para as interações que foram construídas. A utilização das ferramentas do Google Drive foi de suma importância para as solicitações e retornos mais urgentes que surgiram. Da concepção do Processo de SE até a versão final do software utilizamos o recurso da internet para viabilizar interações, tirar dúvidas sobre os requisitos, sobre as percepções dos desenvolvedores, entre outros.

6.2 Observações sobre a utilização da Engenharia Didático-Informática / Considerações sobre o Processo de desenvolvimento do software

Como objetivo geral, pretendíamos construir, analisar e validar um Processo de Desenvolvimento de Softwares Educativos, sendo observados os métodos da engenharia de requisitos na Engenharia de Softwares integrados com a Engenharia Didática, articulados em uma análise de potencialidades teóricas e tecnológicas. Acreditamos que esse objetivo foi alcançado, no momento em que o processo desenvolvido foi validado no estudo de caso proposto. Ao desenvolvermos o SE pretendido, conseguimos aprimorar o que fora proposto no processo de software que estava em construção chegando a uma versão final desse processo.

Os objetivos específicos também foram alcançados visto que propomos realizar um uma análise de metodologias de desenvolvimento de SE existentes, identificando potencialidades e limitações. Propusemo-nos também a articular elementos das teorias da Didática da Matemática e da Engenharia de software para desenvolver um processo de Engenharia de Software e; estudar um caso de desenvolvimento de um software seguindo o processo criado; e assim o fizemos.

Contudo, algumas situações merecem destaque e servem para encaminhamentos futuros das próximas versões que serão desenvolvidas para o aperfeiçoamento da Engenharia Didático-Informática aqui discutida. As sessões seguintes descrevem alguns desses encaminhamentos.

6.2.1 Definição da interface

Consideramos que a interface é uma característica que permearia as dimensões cognitivas e informáticas, entretanto não contemplamos em nossa análise características pertinentes a interface do software. A análise realizada para o desenvolvimento do software levou em consideração a interface do ponto de vista da contribuição para a construção de conhecimentos, não do ponto de vista de teorias cognitivas ou tecnológicas sobre a concepção de interfaces.

Contudo, mesmo com a ausência de contribuições teóricas sobre a definição de interface, ao serem considerados os métodos ágeis no desenvolvimento houve a construção do ambiente de modo a favorecer rápidas alterações e solicitações diversas que pudessem ser feitas, propiciando assim agilidade nos feedbacks que surgiam.

O layout seguiu padrões de ambientes de simulação, onde botões, menus, controladores e as demais funcionalidades são dispostos na tela principal e podem ser acessadas facilmente pelo usuário. Ainda assim, acreditamos que para uma próxima versão do processo de desenvolvimento de software discutido nesta pesquisa devem ser integrados os conhecimentos sobre definição de layout e interface de uma forma geral.

6.2.2 Equipe pluridisciplinar

A partir da revisão de literatura e análise do referencial teórico, percebemos a importância da constituição de uma equipe pluridisciplinar para a criação de softwares educativos. Essa equipe, com uma metodologia de trabalho que permita a integração, fornece conhecimentos de diversas áreas para a concepção, elaboração e utilização do que se pretende criar. Nossa proposta de desenvolvimento considera que essa equipe deve ser montada com diversos profissionais, a fim de obter conhecimentos de áreas distintas que possam contribuir para o produto a ser criado.

Para o software que foi criado, a equipe de desenvolvimento contribuiu com suas experiências de ensino, com os referenciais teóricos estudados em suas pesquisas, com a indicação de possíveis modificações no processo de software e sugerindo requisitos e implementações no software quando em seu desenvolvimento.

Além disso, uma variável que deve ser levada em consideração é o conhecimento do engenheiro-programador sobre tecnologias para a Educação Matemática. Acreditamos que

outro profissional, que não tivesse esses conhecimentos, não teria compreendido tão facilmente os requisitos levantados nas análises teóricas. Com isso, confirmamos a necessidade da constituição de uma equipe pluridisciplinar, tanto para a compreensão das particularidades dos conhecimentos matemáticos, quanto na transposição destes conhecimentos para os artefatos tecnológicos. Tal transposição, em nosso caso, foi favorecida pela experiência e conhecimentos que o programador possui na área da Tecnologia e Educação Matemática.

Quanto à composição da equipe pluridisciplinar, registramos que algumas incompreensões na Dimensão Cognitiva podem ter sido ocasionadas pela falta de um profissional da área da Psicologia.

Contemplamos na equipe, profissionais com formação, qualificação, conhecimentos e experiências nas Dimensões Didática, Epistemológica e Informática. Porém, acreditamos que um profissional com formação em psicologia poderia trazer significativas contribuições para o desenvolvimento do processo de software e o produto criado. Contudo, o acesso a área cognitiva foi possibilitado não com os profissionais especialistas, mas com o recurso da análise teórica ao que se tem discutido nesse campo de estudos.

Além disso, como se observou na sessão anterior, os elementos de definição do layout da interface, do ambiente construído, poderiam ser indicados por um profissional de design e arquitetura de softwares.

6.2.3 Fase experimental e Teoria da Orquestração instrumental

A fase experimental de um protótipo de software é de essencial importância para determinar elementos fundamentais à formulação e reformulação do produto de software em desenvolvimento. Isto porque esta fase pode revelar elementos que não foram contemplados na concepção, seja nas análises teóricas, no mapeamento de requisitos, ou no levantamento das situações de utilização.

Desse modo, analisamos a Teoria da Orquestração Instrumental – TOI (TROUCHE, 2004; 2005; 2009) e percebemos que este pode ser um aporte teórico-metodológico que contribui quanto à criação de situações de utilização do software educativo, delimitando uma série de descritores a serem observados no processo de experimentação software. Trouche (2005) formalizou a noção de orquestração para modelizar a atividade de elaboração e

gerenciamento de situações com integração de tecnologias por parte do professor – Escola de Altos Estudos (2015). De acordo com Trouche (2009),

> Uma orquestração instrumental é a organização sistemática dos artefatos disponíveis em um determinado ambiente, estabelecendo o trabalho de uma determinada atividade matemática. A atividade matemática pode passar por várias fases: a descoberta do problema, a pesquisa individual, o trabalho em grupo, a partilha, o retorno reflexivo sobre a atividade. (TROUCHE, 2009. p. 36).

O trabalho de elaboração de uma situação ou sequência de ensino por um professor, mais ainda quando se trata de integrar tecnologias computacionais, pode ser assimilado a um trabalho de engenharia. Esta prática vem complementar a engenharia já realizada para a concepção e o desenvolvimento das tecnologias disponíveis. Neste processo, a inserção da TOI ao quadro teórico da Engenharia Didático-Informática poderá render análises importantes para a criação e observação de situações de uso dos SE que estão sendo desenvolvidos.

Com isso, a *Fase Experimental* do processo criado pode ser futuramente aprimorada utilizando pressupostos teóricos e metodológicos da Teoria da Orquestração Instrumental – (TROUCHE, 2009), pois o mapeamento de variáveis que a TOI possibilita é capaz de apresentar, mesmo na fase de experimentação do software, contribuições para a concepção do produto. Percebe-se que o referencial da TOI é fundamental para o desenvolvimento e análise de situações didáticas com a utilização de artefatos tecnológicos. Com isso, é de grande importância inserir princípios dessa teoria na fase de validação de um SE, acredita-se que ocorrerão contribuições significativas para a formulação e reformulação dos produtos que se pretendem criar.

Outra hipótese considerada neste texto foi que a utilização de novas tecnologias por si só não é suficiente para uma melhora significativa do ensino e da aprendizagem de conhecimentos. Em resposta a essa hipótese observa-se em Trouche (2004) a necessidade da organização formal, bem como um planejamento prévio, para a criação de situações que viabilizem a aprendizagem matemática com artefatos, sejam eles tecnológicos ou não. De acordo com ele, a orquestração é decomposta em dois elementos essenciais: a configuração didática e o modo de exploração,

> Uma orquestração instrumental é definida pelas *configurações didáticas* (ou seja, o layout dos artefatos disponíveis no ambiente, com um layout para cada etapa do tratamento matemático) e pelo *modo de exploração* dessas configurações e aponta a necessidade de direção externa na gênese instrumental que estão imbuídos dentro da coordenação de um conjunto de instrumentos e na organização dos sujeitos envolvidos (professores, aluno e tempo) no ambiente de trabalho e/ou estudo. A Gênese Instrumental, é um "processo complexo, e ligada às características dos

> artefatos (suas *potencialidades* e *restrições)* e à atividade do sujeito, sua / seu conhecimento e método de trabalho" (TROUCHE , 2004, p. 285, tradução nossa).

A partir dessa verificação e ao se analisar a fase experimental proposta no Modelo de Processo de *Software* observamos na TOI uma fonte de recursos teóricos e metodológicos para uma análise que apresente elementos da utilização dos produtos que se pretende criar. De acordo com Trouche (2009) uma Orquestração Instrumental é a organização sistemática dos artefatos disponíveis em um determinado ambiente, estabelecendo o trabalho de uma determinada atividade matemática. Essa organização, pré-determinada, irá favorecer na observação de elementos de análise para o desenvolvimento dos softwares concebidos através deste modelo de processo.

Com isto, percebemos que, para uma nova versão do processo de desenvolvimento de SE que foi concebido neste estudo, a utilização da Engenharia Didática e da Teoria da Orquestração Instrumental em diferentes fases, porém articuladas, podem aperfeiçoar as próximas versões.

REFERÊNCIAS

ABREU, F., ALMEIDA, A., BARREIROS, E. SARAIVA, J.,SOARES, S., ARAUJO, A., HENRIQUE G. Métodos, Técnicas e Ferramentas para o desenvolvimento de Software Educacional: Um Mapeamento Sistemático. In: 23º SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 23. 2012, Rio de Janeiro. **Anais...** Rio de Janeiro: Sociedade Brasileira de Computação, 2012.

AGILE ALIANCE. **Manifesto for Agile Software Development**. Disponível em: <http://agilemanifesto.org>. Acesso em: 01 fev. 2015.

ALMOULOUD, S. A.. Didática e concepção de dispositivos informáticos educacionais. **Revista de Informática Aplicada**, São Caetano do Sul, v. 3, n. 1, 03-10, Jan/Jun 2007.

ARTIGUE, M. Engenharia Didática. In: BRUN, J. **Didática das Matemáticas**. Lisboa: Instituto Piaget, 1996. Cap4, p. 193 -217.

BAIRRAL, M. A. Do clique ao touchscreen: Novas formas de interação e de aprendizado matemático. In: 36A REUNIÃO NACIONAL DA ANPED, 36, 2013, Goiânia. **Anais...**Goiânia: Anped/UFG, 2013.

BALACHEFF, N.; BELLEMAIN, F.; Conhecimento, a Pedra Angular do Design de Tel. **Tópicos Educacionais**, v. 17, p. 31-59, 2007.

BELLEMAIN, F.. O Paradigma Micromundo. In: COLÓQUIO DE HISTÓRIA E TECNOLOGIA NO ENSINO DE MATEMÁTICA HTEM, 2002, Rio de Janeiro. **Anais...** Rio de Janeiro: UFRJ, 2002, pp. 51-62.

BELLEMAIN. F.; BELLEMAIN, P. M. B.; GITIRANA. V. Elementos de engenharia de software educativos para a concepção de ferramentas computacionais para o CSCL. In ROSA, M.; BAIRRAL, M. A.; AMARAL, R. B. **Educação Matemática, Tecnologias Digitais e Educação a Distância: pesquisas contemporâneas**. Natal (RN): Editora da Física, 2014.

BELLEMAIN, F.; RAMOS, C. S.;TIBÚRCIO, R S. Engenharia de software educativos, o caso do bingo dos racionais. In: 6 SEMINÁRIO INTERNACIONAL DE PESQUISA EM EDUCAÇÃO MATEMÁTICA - SIPEM, 2015, Goiás. **Anais do VI Sipem...** Goiás: SBEM, 2015.

BENITTI, F. B. V., SEARA, E. F. R., SCHLINDWEIN, L. M. Processo de Desenvolvimento de Software Educacional: Proposta e Experimentação. **Revista Novas Tecnologias na Educação – CINTED UFRGS** v. 3, n. 1, Maio, 2005.

BRASIL. Secretaria de Educação Fundamental. **Parâmetros Curriculares Nacionais: terceiro e quarto ciclos do Ensino Fundamental**: Matemática. Secretaria de Educação Fundamental. Brasília: MEC/SEF, 1998a.

BRASIL. Secretaria de Educação Média e Tecnológica. **Parâmetros Curriculares Nacionais – Ensino Médio — PCNEM**. Brasília: MEC, 1998.

BRASIL. Secretaria de Educação Básica. **Orientações Curriculares para o Ensino Médio – Ciências da natureza, Matemática e suas Tecnologias**. Brasília: MEC; SEB, 2006.

BRASIL Ministério da Educação. **Portaria nº 522/MEC, de 09 de abril de 1997**. Disponível em:<http://www.anatel.gov.br/hotsites/direito_telecomunicacoes/textointegral/nor/prt/med_19970409_522.pdf>. Acesso em: 15 de setembro 2011.

CONFORTO E. C., AMARAL D. C., SILVA S. L. Roteiro para revisão bibliográfica sistemática: aplicação no desenvolvimento de produtos e gerenciamento de projetos. In: *CONGRESSO BRASILEIRO DE GESTÃO DE DESENVOLVIMENTO DE PRODUTOS*, 8., 2011, Porto Alegre. **Anais...** Porto Alegre: UFRGS, 2011. p. 1–12.

COSTA, A. P. **Metodologia Híbrida de Desenvolvimento Centrado no Utilizador**. Tese de doutorado. Universidade de Aveiro, Aveiro, 2012.

COSTA, A. P., COSTA, E. B. Contributos para o Desenvolvimento de Software Educativo tendo por base Processos Centrados no Utilizador. **EM TEIA | Revista de Educação Matemática e Tecnológica Iberoamericana**, v. 4, n. 2, p. 1–15, 2013.

ESCOLA DE ALTOS ESTUDOS. Home. Disponível em: http://lematec.net.br/EAE/index.html?page=inicial&language=br. Acesso em 20 set. 2015.

GALVIS-PANQUEVA, A.H. Software Educativo Multimídia: Aspectos Críticos no seu Ciclo de Vida. **Revista Brasileira de Informática na Educação**, Comissão Especial de Informática na Educação da Sociedade Brasileira de Computação, Florianópolis, SC, Setembro, p. 9-18, 1997.

GARCIA, V. C.; MACHADO, C. Teorias de pesquisa em Educação Matemática: a influência dos franceses. Coordenação: Vera Clotilde Garcia. Disponível em: http://143.54.226.61/~vclotilde/disciplinas/pesquisa/CLAUDIA_FRANCESES.D%20OC.pdf. Acesso em: 17 set. 2015.

GIL, A. C. **Como Elaborar Projetos de Pesquisa**. 4ª ed. São Paulo: Atlas, 2002.

GITIRANA, V. Função Matemática: o entendimento dos alunos a partir do uso de softwares educacionais. **A pesquisa em Educação Matemática: repercussões na sala de aula**, in Borba, R. e Guimarães, G. São Paulo: Cortez, 2009.

KAPUT, J. Technology and Mathematics Educacion. In: D.A. GROWS (ED.) HANDBOOK OF RESEARCH ON MATHEMATICS TEACHING AND LEARNING, **Anais…** Macmillan, NY, 1992, p. 515-556.

KITCHENHAM, B., CHARTERS, S. Guidelines for performing Systematic Literature Reviews in Software Engineering. **EBSE Technical Report**. Reino Unido, v. 2.3, 2007. Disponível em: https://www.cs.auckland.ac.nz/~mria007/Sulayman/Systematic_reviews_5_8.pdf. Acesso em 05 set. 2015.

LAPOLLI, F. R. ; MOTTA, C.L.R. ; OLIVEIRA, C. E. T. ; CRUZ, C. M. Modelo de Desenvolvimento de Objetos de Aprendizagem Baseado em Metodologias Ágeis. In: XX SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 20., 2009, Florianópolis. **Anais...** Porto Alegre: Sociedade Brasileira de Computação, 2009. p. 250-261.

LEMOS, A. Cibercultura. Alguns pontos para compreender a nossa época. In: LEMOS, A., CUNHA, P. **Olhares sobre a cibercultura**. 1. Ed. Porto Alegre: Sulina, 2003.

LIMA, M. M.; LIMA, A. R.; MONTEIRO, A. C. C.; CAVALCANTE, E. H.; GOMES, L. Q. L. Uma revisão sistemática da literatura dos processos de desenvolvimento de software educativo. In: XXIII SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 23, 2012, Rio de Janeiro. **Anais...** Rio de Janeiro: UFRJ, 2012.

LUCENA, M. W. F. P. Diretrizes para a Capacitação de Professores na Área de Tecnologia Educacional: Critérios para Avaliação de Software Educacional. **Revista Virtual de Informática Educativa e Educação a Distância**, 1998.

MULBERT, A. L.; PEREIRA, A. T. C.. Um panorama da pesquisa sobre aprendizagem móvel (m-learning). In: V SIMPÓSIO ABCIBER, 5, 2011, Florianópolis. **Anais...**Florianópolis: UFSC/UDESC, 2011. v. 1. p. 1-13.

NUSEIBEH B., EASTERBROOK S. Requirements Engineering: A Roadmap, The Future of Software Engineering. In: 22ND INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, ACM-IEEE, 22.,Irlanda. **Anais…** Irlanda: ICSE, 2000.

PERES, D. R. **Modelo de Qualidade para componentes de *software***. 151 p. Dissertação (Mestrado em Ciências de Computação e Matemática Computacional). USP – São Carlos. Nov/ 2006. Disponível em: http://www.teses.usp.br/teses/disponiveis/55/55134/tde-29012007-112338/pt-br.php.Acesso em 17/04/2013.

PRESSMAN, R. S. **Engenharia de Software**. 6ª edição, McGraw-Hill, 2006.

PORTAL DO LEMATEC. *Home*. Disponível em: <http://www.lematec.no-ip.org/>. Acesso em 28 de janeiro de 2015.

SANTOS, G. L. Alguns princípios para situações de engenharia de softwares educativos. **Inter-ação**, Goiás, v. 34, n. 1, 2009. Disponível em: http://www.revistas.ufg.br/index.php/interacao/article/view/6540/4801. Acesso em: 06 set. 2010.

SILVA, C. T. J. O Uso de Simulações no Ensino de Função: uma sequência didática para abordar taxa de variação. In: XVIII EBRAPEM - ENCONTRO BRASILEIRO DE ESTUDANTES DE PÓS-GRADUAÇÃO EM EDUCAÇÃO MATEMÁTICA, 18., 2014, Recife. **Anais...** Recife: LEMATEC, 2014.

SOMMERVILLE, I. **Engenharia de Software**. São Paulo 8. ed. Pearson Education do Brasil, 2007.

_____, I. **Engenharia de Software**. São Paulo. 6. ed. Pearson Education Companion, 2003.

TCHOUNIKINE, P. Pour une ingénierie des Environnements Informatiques pour l'Apprentissage Humain. **Revue I3 InformationInteractionIntelligence**. v 2, n. 1, p. 59-95. 2002.

TCHOUNIKINE P., BAKER M., BALACHEFF N., BARON M., DERYCKE A., GUIN D., NICAUD J.-F., RABARDEL P.. Platon-1 : quelques dimensions pour l'analyse des travaux de recherche en conception d'EIAH, 2004. Rapport d'ActionSpécifiquedu CNRS.

TIBÚRCIO, R S. Método de Elicitação de Requisitos para Software Educativo: um estudo a partir da prototipação de um software para função em plataformas móveis. In: XVIII EBRAPEM - ENCONTRO BRASILEIRO DE ESTUDANTES DE PÓS-GRADUAÇÃO EM EDUCAÇÃO MATEMÁTICA, 18., 2014, Recife. **Anais...** Recife: LEMATEC, 2014.

TROUCHE, L. Penser la gestion didactique des artefacts pour faire et faire faire des mathématiques: histoire d'un cheminement intellectuel. **L'Educateur**, v. 309, p. 35-38, 2009.

_____. Managing the complexity of human/machine interactions in computerized learning environments: guiding students' command process through instrumental orchestrations. **International Journal of Computers for Mathematical Learning**, 9, 281-307.2004.

_____. Construction et conduit des instruments dans les apprentissages mathématiques: nécessité des orchestrations. **Recherches em Didactique des Mathématiques**. v. 25, p. 91-138, 2005.

APENDICE A – RESPOSTAS AOS QUESTIONAMENTOS

1. DELIMITAÇÃO DO CAMPO

1.1 Qual campo de conhecimentos pretende-se abordar? Dentro deste campo de conhecimentos delimite conceitos e definições que serão trabalhadas e qual o foco que será dado ao conhecimento trabalhado.

O conhecimento a ser abordado é a taxa de variação das principais funções reais de uma variável real, geralmente abordadas no Ensino Médio, bem como de funções obtidas por operações básicas (adição, subtração, multiplicação e quociente) e pela composição com essas funções. O conceito será abordado sob uma perspectiva variacional de função, no contexto da covariação, com base no Quadro de ações e níveis mentais no raciocínio covariacional, de Carlson et al (2002). Nesse sentido os principais conceitos e ideias abrangidos são a variação, a taxa de variação média e a taxa de variação instantânea (como sendo a derivada) das funções reais, abordadas em notações diversas (gráfico, tabela, modelo algébrico, etc) e articuladas com o contexto geométrico da derivada como a inclinação da reta tangente ao gráfico.

1.2 Levando em consideração as tipologias de software educacionais existentes (tutoriais, jogos, simuladores, etc.) qual tipo de software pretende-se criar e quais são as justificativas para essa escolha?

A abordagem do conceito por um ambiente computacional que permita simulações dinâmicas, interativas e em múltiplas representações pode contribuir de forma geral por possibilitar um tratamento de caráter variacional da função e da sua taxa de variação, permitindo aos alunos abordar o conceito com o apoio do computador por ferramentas que auxiliem na análise do que acontece com a taxa na variação conforme se varia as variáveis da função.

1.3 Existem ambientes que trabalham com o domínio de interesse, quais são os recursos que contribuem para a aprendizagem e o que falta nesses softwares para contemplar as necessidades do domínio?

a) *Modellus*: Permite simulações do modelo funcional ao incorporá-lo a diferentes objetos em uma animação. Possibilita representar e descrever a encenação do modelo por diferentes

representações conectadas ao modelo algébrico. A abordagem covariacional é facilitada nesse software pois permite encenações de modelos em que a dependência entre variáveis fica mais clara em "gráficos animados", onde a curva vai sendo traçada conforme se varia a variável independente ou algum parâmetro da função. Existem alguns pontos a melhorar no Modellus em relação ao conceito da taxa, um deles é a articulação entre as representações e as possibilidades de ação sobre o gráfico principal e a tabela, que não permitem manipulações diretas sobre eles o que caracteriza-os mais como notações de exibição do que de ação. Outro ponto é que de forma geral, as simulações dependem do tempo como variável independente, o que restringe o modelo a essa variável.

b) *Geogebra*: Articula a álgebra e a geometria dinâmica, com múltiplas representações conectadas dinamicamente. O modelo algébrico pode ter seus coeficientes conectados a controles deslizantes, aumentando o dinamismo. A tabela (planilha) permite ações como criar funções para relacionar variáveis (semelhante às planilhas eletrônicas usadas atualmente), o que permite uma abordagem numérica dinâmica da covariação entre variáveis, além da conexão ao gráfico e ao modelo algébrico, o que facilita a abordagem da taxa de variação. O GeoGebra é um software bastante completo por possibilitar uma abordagem covariacional dinâmica com múltiplas notações, porém o que restringe o software no aspecto funcional é justamente o fato de não ser um software com foco a priori em funções, e sim na articulação álgebra-geometria dinâmica, dessa forma aspectos puramente funcionais podem não ter maior foco no ambiente, como a definição do domínio ou de um intervalo específico, a dependência entre variáveis ou a construção do gráfico variacionalmente.

c) *Winplot*: Traça o gráfico da função a partir do modelo algébrico, permite dinamismo de forma limitada ao conectar coeficientes do modelo algébrico a controles deslizantes para um intervalo discreto. A representação tabular é apenas para fins de exibição dos valores das variáveis, não é permitida ação direta sobre o gráfico ou a tabela, apenas a partir do modelo algébrico. Como a função principal do software é traçar o gráfico, não há conexão com outros sistemas como a geometria dinâmica ou a física, por isso as simulações são bastante restritas, dificultando uma abordagem covariacional.

d) *Casyopée:* Modela relações funcionais a partir da geometria dinâmica, articula múltiplas representações de função, de forma dinâmica e conectada, partindo da modelagem da situação no contexto geométrico, gerando o modelo algébrico e conectando às demais representações. Além disso, permite o tratamento do modelo algébrico por um CAS (computer algebra system). A proposta do Casyopée é de uma perspectiva covariacional, dinâmica e articulada à geometria, o que valoriza o seu potencial principalmente para a abordagem da taxa instantânea ou a derivada. Para a abordagem de taxa de variação, falta uma melhor articulação da representação tabular com as demais representações e as possibilidades de manipulação direta sobre essa representação e sobre o gráfico.

De modo geral, como esses softwares não foram produzidos com o foco na abordagem da taxa de variação das funções, faltam-lhes ferramentas específicas que podem auxiliar e facilitar a atividade dos alunos nesse conceito em um contexto covariacional.

2ANÁLISES PRELIMINARES

2.1 Existem indicações na literatura que contribuam para o desenvolvimento do conhecimento que será trabalhado no software relacionadas a área cognitiva?

Os caminhos apontados pelas pesquisas como potencialmente facilitadores da aprendizagem da taxa de variação vão nas seguintes direções:

a) Abordagem do conceito articulando e conectando diferentes representações relacionadas à taxa de variação, de forma dinâmica, o que supõe o uso de um ambiente que propicie a dinamização nessas representações.

b) Abordagem do conceito por uma perspectiva variacional, no contexto da covariação e de forma dinâmica, que pode contribuir com a compreensão da taxa instantânea, cuja compreensão está relacionada à ideia de limite.

2.2Qual é o estado atual do ensino do domínio? Quais são as consequências desse ensino?

Abaixo são listados alguns aspectos do ensino do conceito que devem ser considerados ao analisar o ensino e a aprendizagem de taxa de variação:

a) Abordagem do conceito em “mídias estáticas”: requer do aluno um esforço mental muito grande para visualizar a variação, um aspecto dinâmico.

b) Abordagens que focalizam no tratamento algébrico: limita uma visão variacional e fortalece uma visão pontual da função.

c) No Ensino Médio se define a taxa algebricamente em termos de funções específicas, já no Ensino Superior revisita como uma interpretação da derivada a partir da noção de limite e desvinculada de funções específicas: as abordagens têm focos distintos, no Ensino Superior o conceito de limite é fundamental.

d) Problemas na articulação com o contexto geométrico: aspectos como a articulação com a inclinação da reta tangente podem gerar dificuldades quando o conceito da tangente é simplesmente importado da geometria plana e não contextualizado no Cálculo e a inclinação dessa reta sofre mudanças visuais com as mudanças das escalas do gráfico, o que também traz problemas com a associação da noção de coeficiente angular da reta e a taxa de variação.

2.3Quais são as características do conhecimento que dificultam a aprendizagem?

Abaixo são listados alguns aspectos próprios do conceito que devem ser considerados ao analisar o ensino e a aprendizagem de taxa de variação:

a) Interpretação gráfica da taxa de variação: necessidade de fazer uma leitura variacional do gráfico e relacionar concavidade e pontos de inflexão com a taxa de variação.

b) Variação da taxa de variação: a taxa de variação determina a variação da função, mas a própria taxa pode ser constante ou variar, quando ela varia o aluno precisa lidar com a variação da variação.

c) Sinal da taxa de variação: a taxa pode ser positiva ou negativa, foi relatada dificuldade dos alunos em interpretar a taxa quando negativa.

d) Passagem da taxa média para a instantânea ou a derivada – relação com limite: o conceito de limite torna-se fundamental na compreensão da passagem da taxa média para a instantânea, logo é preciso compreender esse conceito para compreender tal passagem.

e) Articulação com a geometria: A passagem da taxa média para a derivada é fortemente articulada com a geometria no sentido de uma “aproximação” da reta tangente ao ponto por retas secantes a dois pontos, essa articulação traz em si questões como a importação do

conceito de reta tangente da geometria plana que deve ser adaptado ao Cálculo. Além disso, as associações entre a derivada e o coeficiente angular da reta tangente, bem como os aspectos visuais do gráfico como a inclinação da reta e as mudanças de escalas do gráfico também merecem atenção na articulação com a geometria.

2.4 Quais as características fundamentais que o ambiente deve conter para atender as necessidades/características da aprendizagem do domínio?

Para se enquadrar nos requisitos que, segundo sugerem as pesquisas, são potencialmente favoráveis à aprendizagem do conceito, o ambiente deverá conter de forma geral as características abaixo:

a) Perspectiva covariacional

b) Dinamismo

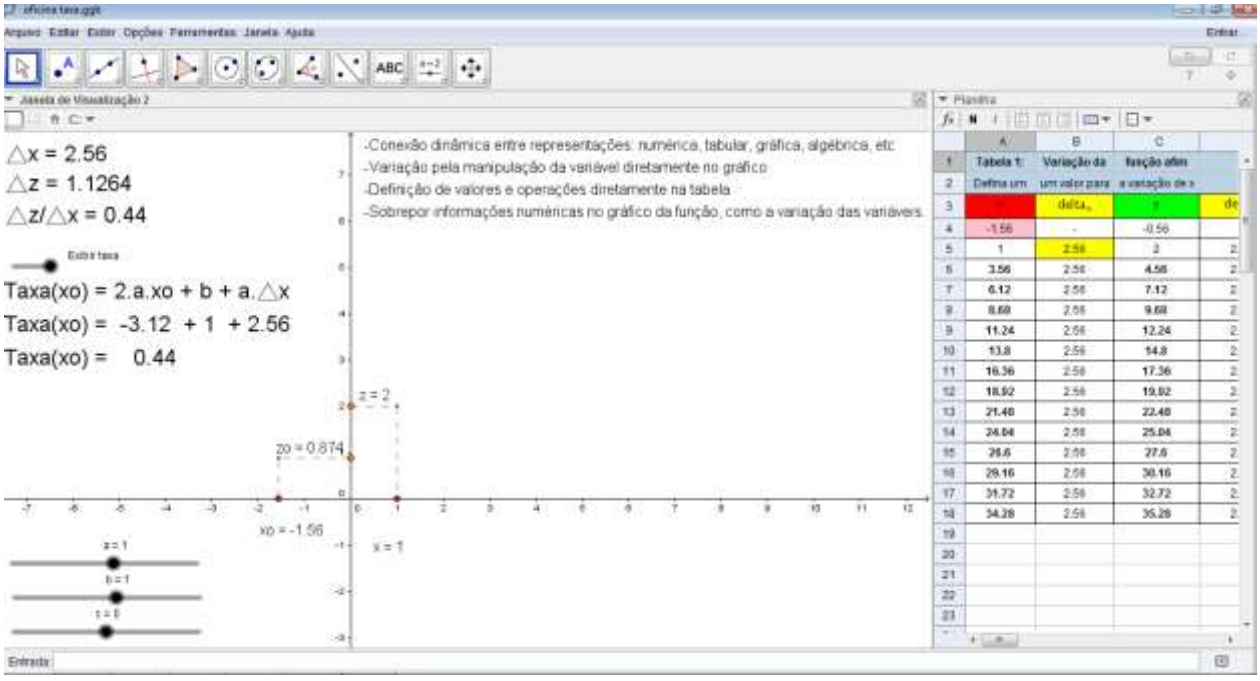
c) Interatividade

d) Conexão dinâmica de múltiplas representações relacionadas à taxa de variação

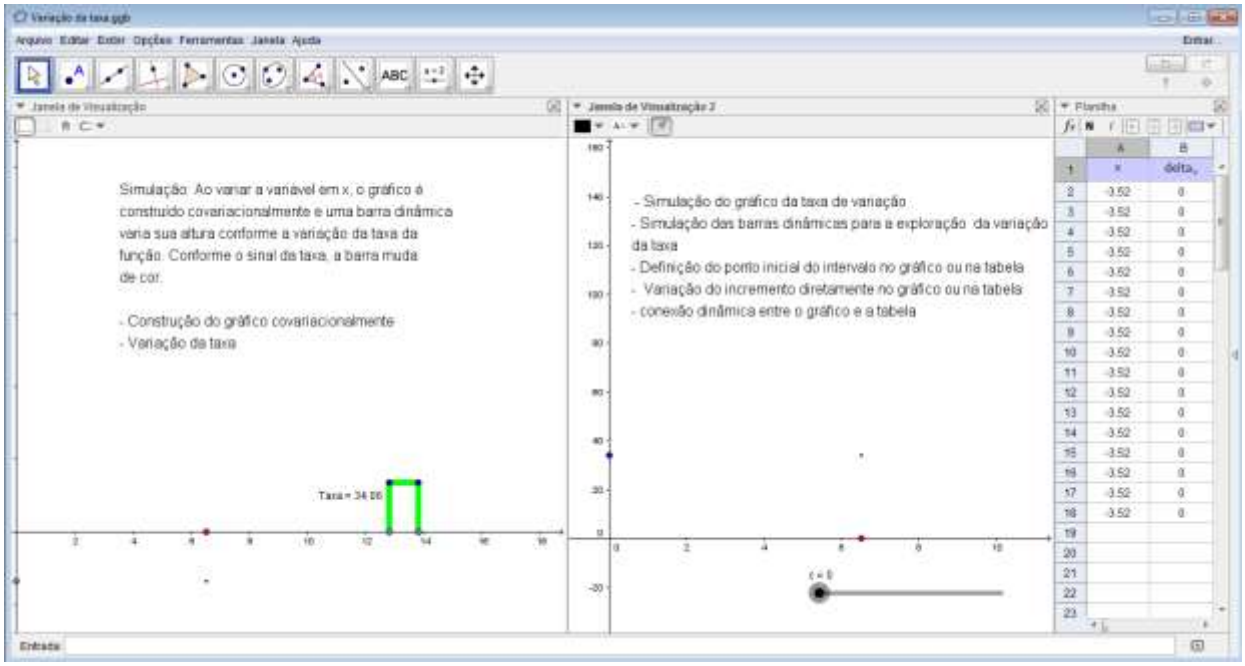
e) Suporte ao raciocínio dos alunos por meio de ferramentas inseridas nos contextos acima

APENDICE B – CONSTRUÇÕES UTILIZANDO O GEOGEBRA

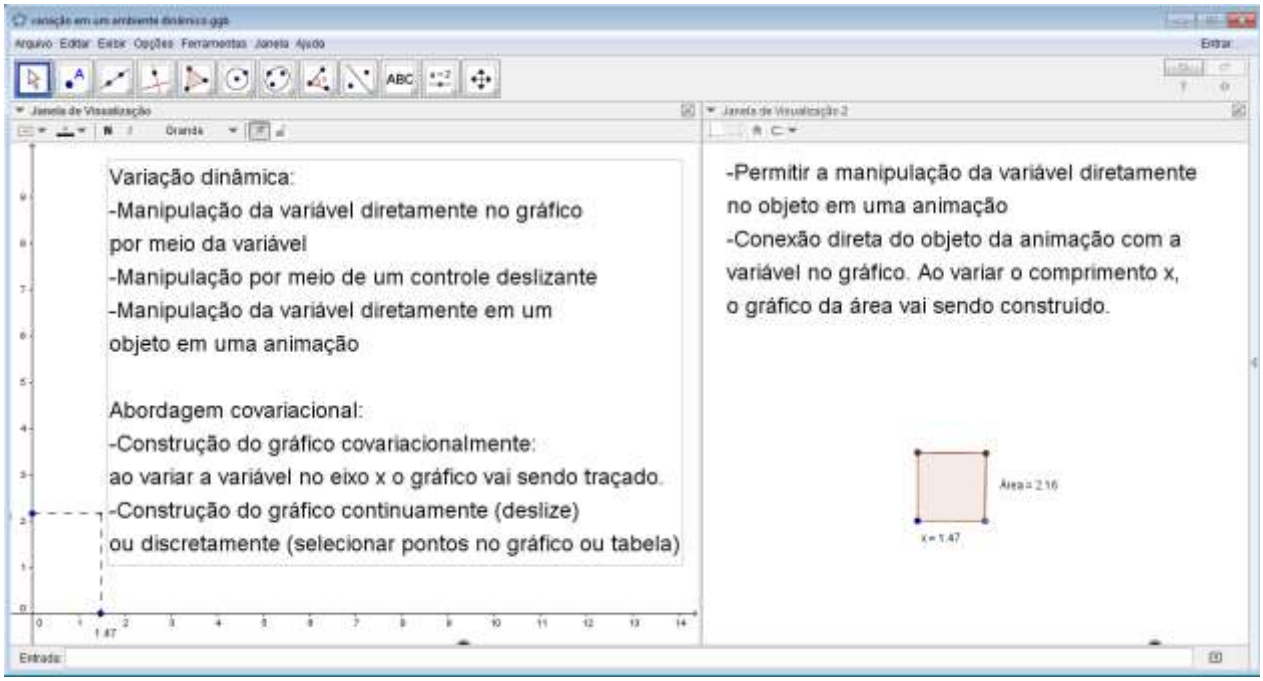
a) Múltiplas representações



b) Taxa de variação



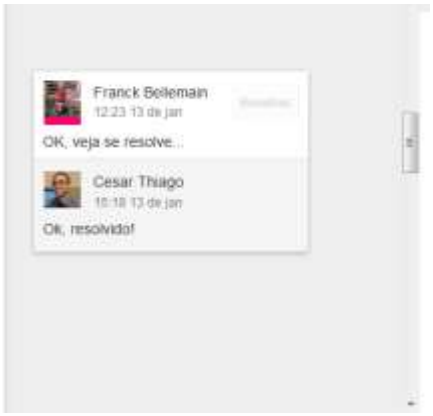
c) Variação em um ambiente dinâmico



APÊNDICE C – RETORNO A PRIMEIRA VERSÃO DO SOFTWARE

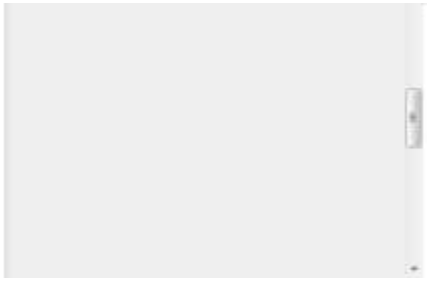
a) Ferramenta taxa de variação

- **Ferramenta taxa de variação**
  - **Definição do intervalo:** Ao definir o intervalo no qual será calculada a taxa de variação no gráfico, exibir a informação numérica dos pontos P0 e P1 (atualmente só são exibidas as coordenadas de P0)
  - **Exibição do valor numérico da variação de y no gráfico**: Quando é aplicada a ferramenta taxa de variação no gráfico, um triângulo (de delta y e delta x) é representado no intervalo tomado, este triângulo varia ao variar o ponto P. É interessante que quando se estiver variando o ponto em x, seja exibido o valor de "delta y" variando simultaneamente.
  - **Partição:** Na ferramenta taxa de variação, permitir que um intervalo seja particionado em "deltas x" iguais proporcionando uma análise da variação em uma sequência. As informações numéricas a serem exibidas no gráfico podem se limitar às coordenadas do ponto inicial(P0) e a do ponto final, evitando várias informações na tela. Os demais pontos do intervalo podem ser configurados para exibir seus valores ao posicionar-se o cursor em cima do ponto.



b) Ferramenta limite

- **Ferramenta limite**
  - Ao escolher essa ferramenta e uma função dada, permitir que na aproximação entre dois valores no eixo x (x e x0), o gráfico exiba a aproximação entre os dois valores correspondentes no eixo y(y e y0), simulando o limite. Seria interessante que nessa ferramenta se usasse o zoom dinâmico, mas as ferramentas de zoom que foram implementadas nos eixos x e y já permitem uma primeiro passo.
  - É interessante também que tal ferramenta possa permitir a abordagem dos limites laterais.



c) Ferramenta reta tangente

- **Ferramenta reta tangente**
  - É interessante que o software contemple também retas secantes ao gráfico, para abordar as situações da "aproximação" da tangente por secantes.

d) Ferramenta tabela

- **Tabela:**
    - **Possibilidade de edição da tabela:** Representar os valores da função em uma tabela e atrelar a variação no gráfico à variação na tabela. Além disso, permitir definir uma novas colunas, como uma coluna que calcule a taxa de variação entre dois valores da função ou uma coluna que calcule a variação entre os valores da coluna anterior.

APÊNDICE D – PROGRAMAÇÃO DA PRIMEIRA VERSÃO

Os critérios para a escolha do alvo e da linguagem foram essencialmente:

- multiplataforma;

- simplicidade da implementação.

Nesse sentido, optamos por desenvolver o artefato para plataforma web, utilizando as linguagens *HTML* (*HTML5*), *CSS* e *JAVASCRIPT*. Essas linguagens, sendo interpretadas por padrão por qualquer browser, podem ser editadas com um simples tratamento de texto e beneficiam-se de inúmeras bibliotecas de objetos permitindo encurtar o tempo de desenvolvimento e de se dedicar, quase exclusivamente, na implementação do código dos artefatos que diz respeito à transposição didática-informática do conceito de função e de taxa de variação e à proposta didática de ensino desses conceitos.

A escolha de uma prototipagem para plataforma web facilita também a distribuição do artefato para os diversos envolvidos no projeto e favorece as reconfigurações no processo de engenharia de software educativo, entre a engenharia de software empregada para a concepção/desenvolvimento técnico do artefato e a engenharia de requisitos que fundamenta e norteia essa concepção/desenvolvimento. O emprego de métodos ágeis ou de uma metodologia que favoreça uma forte e rápida interação entre os diversos profissionais envolvidos no projeto é facilitada pela escolha de desenvolvimento web.

Modelo MVC

Mesmo se o desenvolvimento do artefato utiliza tecnologias da web, não se trata de um desenvolvimento clássico cliente-servidor onde a metodologia *MVC* aplica-se frequentemente, a combinação das linguagens utilizadas: *HTML*, *CSS* e *JAVASCRIPT*, favorece a separação na concepção/desenvolvimento do artefato entre o modelo (M), o controlador (C) e a visualização (V).

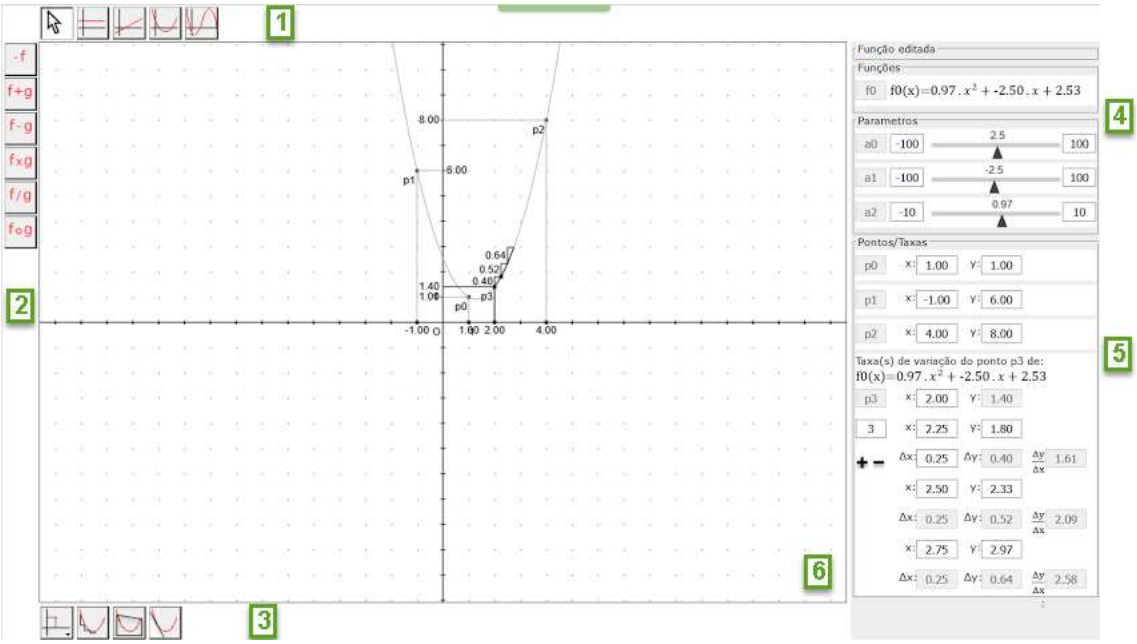
A camada VIEW

O *HTML* junto com *CSS* permite a elaboração do layout do artefato independente do comportamento dos elementos *HTML* que compõem esse modelo. O *javascript* permite a programação da interação entre os elementos que compõem o modelo (e mais especificamente os objetos matemáticos representados nesses elementos) e o usuário. Essa interação especifica

ocorre por meio dos objetos de tipo *CANVAS* que junto com o *javascript* permite simular manipulações gráficas pelo mouse ou pelo *touchscreen* de representações dos objetos matemáticos em jogo.

O layout construído para essa primeira versão do protótipo é apresentado na ilustração seguinte:

Layout da primeira versão



1: barra de ferramentas para a construção de funções por família. Na versão atual, temos as funções politomias de grau 0, 1, 2 e 3.

2: barra de ferramentas para a construção de funções como resultado de operações entre funções. Na versão atual, temos a possibilidade de definir a função oposta de uma função, as funções soma, diferencia, produto, razão e composição de duas funções.

3: barra de ferramentas para a construção de elementos específicos relativos a funções assim como um ponto do grafo de uma função cujo coordenadas satisfazem a equação y=f(x), a taxas de variação de uma função, secantes, tangentes.

4: elemento de html onde ficam destacadas a função, os coeficientes da sua expressão, valores particulares, etc. que esta sendo definida ou editada depois de selecionada pelo usuário.

5: elemento de html onde são listas os diferentes elementos construídos: funções, coeficientes de expressões, pontos e taxas de variações.

6: CANVAS onde são representadas graficamente os diversos elementos definidos: pontos, funções, taxas de variações, retas secantes e tangentes.

O código *HTML* estruturando o layout da figura acima é apresentado na figura seguinte, onde aparecem os diversos elementos numerados descritos acima.

Elementos numerados

```
<div class='toolbar'><ul></ul></div>\ 1
<div class='sidebar'><ul></ul></div>\ 2
<canvas class='graphzone'></canvas>\ 6
<div class='defzone'>\
    <fieldset>\                                                    4
        <legend>Fun&ccedil;&atilde;o editada</legend>\
        <div class='selection'>\
            <div class='equations'></div>\
            <div class='parameters'></div>\
            <div class='valuestable'></div>\
        </div>\
        <div class='deffuncbuttons'></div>\
    </fieldset>\
    <div class='analytic'>\                                        5
        <fieldset>\
            <legend>Fun&ccedil;&otilde;es</legend>\
            <div class='equations'></div>\
        </fieldset>\
        <fieldset>\
            <legend>Parametros</legend>\
            <div class='parameters'></div>\
        </fieldset>\
        <fieldset>\
            <legend>Pontos/Taxas</legend>\
            <div class='points_tx'></div>\
        </fieldset>\
    </div>\
</div>\
<div class='bottombar'><ul></ul></div>\ 3
```

A camada Model

Com a linguagem *javascript* não dispomos de uma linguagem orientada a objeto propriamente dita. Entretanto, é possível com as noções de função e de protótipo criar estruturas embarcando variáveis e métodos e permitindo de beneficiar da noção de herança entre objetos. Nesse contexto, modelizamos os diversos objetos matemáticos manipulados pelo artefato, particularmente as funções, como subclasse de uma classe inicial *FunctionObject* declarando os campos e métodos comuns à modelização interna desses objetos.

Como descrito na figura seguinte, a estrutura *FunctionObject* embarca como campos comuns a todos os objetos, um nome e uma cor, e como métodos, funções para determinar a imagem de um valor (*imageof*), o tipo (família) da função como por exemplo: "funçao|polinomio|segundograu", ou a representação na linguagem *mathML* da expressão algébrica da função. A partir dessa estrutura inicial (classe), são definidas subestruturas (subclasses), implementando certos métodos de forma especifica a cada família de função. É o caso, por exemplo, do método *imageof* que calcula a imagem de um valor dado e é especifico a cada família de função.

Estrutura *FunctionObject*

```
function FunctionObject(maindiv) {
    this.name="o";
    this.color='#000000';
    this.maindiv=maindiv;
}
FunctionObject.prototype = {
    typeofobject: function(ty){
    isofType: function(ty){
    imageof: function(x){
    isCoefof: function(o) {
    draw: function(context){
    addDefElement: function(p){
    isDefElementof: function(o) {
    calculate: function(objs) {
    closeTo: function(canvp,eixop){
    moveObjTo: function(canvp,eixop,info){
    toMathML: function(wcoefname,wx) {
    addDisplayDiv: function(container,editname) {
    removeDisplayDiv: function(d) {
}
```

Nem todos os métodos da estrutura *FunctionObject* possuem uma implementação efetiva. Os principais métodos implementados são:

- *typeofobject* e *isoftype* (figura abaixo) tratando do tipo dos *FunctionObject*.

Tipo dos FunctionObject

```
typeofobject: function(ty){
    return ty;
},
isofType: function(ty){
    if(this.typeofobject().search(ty)>-1)
        return true;
    return false;
},
```

- *addDisplayDiv* e *removeDisplayDiv*, representados na próxima ilustração, que associa ou desassocia um objeto *HTML* (do tipo *DIV*) ao *FunctionObject*. Considerando a estrutura *FunctionObject*, apresentada aqui como o objeto teórico "função" (mais precisamente trata-se da representação interna do mesmo), o *CANVAS* citado acima é o espaço comum para a representação gráfica dos *FunctionObject*, e os objetos *HTML* associados servem para apresentar outras representações (simbólicas) dos objetos assim como

a expressão algébrica, tabela de valores, coordenadas de pontos, etc. É importante destacar que os elementos *HTML*, seja o *CANVAS* ou os *DIVs*, são construídos para visualizar diversas representações das funções (interface *OUT*), mas podem ser utilizados como interface de manipulação dos objetos (interface *IN*). Retomando a terminologia de Balacheff (1999) e Bellemain (2002), esses *DIVs* representam o sistema fenomenal e o sistema de interpretação.

addDisplayDiv e removeDisplayDiv

```
addDisplayDiv: function(container,editname) {
    if(typeof this.displaydiv == "undefined" || this.displaydiv == null)
        this.displaydiv=new Array();
    var equdiv=document.createElement('div');
    this.displaydiv[this.displaydiv.length]=equdiv;
    equdiv.className = 'parameter';
    equdiv.associatedobject=this;
    var nam = document.createElement('input');
    nam.className='param_name';
    nam.value=this.name;
    if(!editname)
        nam.disabled=true;
    equdiv.appendChild(nam);

    if(typeof container != "undefined" && container != null)
        container.appendChild(equdiv);

    return equdiv;
},
removeDisplayDiv: function(d) {
    if(typeof this.displaydiv != "undefined" && this.displaydiv != null) {
        var nwequdivs=new Array();
        for(var i=0;i<this.displaydiv.length;i++)
            if(this.displaydiv[i]==d)
                this.displaydiv[i].parentNode.removeChild(this.displaydiv[i]);
            else
                nwequdivs.push(this.displaydiv[i]);
        this.displaydiv=nwequdivs;
    }
}
```

- *addDefElement* e *isDefElementof* (figura seguinte) que tratam das relações entre os diversos elementos construídos. Por exemplo, uma função polinomial é em relação com os coeficientes que definem a sua equação. Uma função f é em relação (de incidência) com pontos P(x;y) que satisfazem a equação y=f(x) (pertencendo ao gráfico da função). Essas relações são importantes desde que permitam "recalcular" os objetos dependendo de mudanças feitas pelo usuário no *CANVAS* de representação gráfica ou nos *DIVs* das outras representações (deslocamento de algum ponto do *CANVAS*, edição do valor numérico de um coeficiente, etc.): recalcular os coeficientes de uma equação em função do deslocamento de

algum ponto do grafo da função, recalcular a ordenada de um ponto em consequência da mudança do valor de um coeficiente da expressão algébrica da função.

Relações entre os diversos elementos construídos

```
addDefElement: function(p){
    if(typeof this.defelements == "undefined")
        this.defelements=new Array();
    this.defelements.push(p);
    if(typeof this.defelmts2coef != "undefined")
        this.defelmts2coef();
    return true;
},
isDefElementof: function(o) {
    if(typeof this.defelements != "undefined" && this.defelements.length>0) {
        for(var i=0;i<this.defelements.length;i++) {
            if(this.defelements[i]==o)
                return true;
        }
    }
    return false;
},
```

Dessa estrutura *FunctionObject*, declinamos diversas sub-classes, as principais sendo as classes *NumValue* representando um valor numérico manipulável pelo usuário, *PointObj* representando um ponto do plano e *FunctionReal* (próxima ilustração) representando no artefato de forma genérica qualquer função de uma variável. Os métodos de *FunctionReal* são herdados de *FunctionObject*, acrescentando algumas especificações. É o caso, por exemplo, das funções *delelmts2coef* e *coef2defelmts* que calculam, para a primeira, os coeficientes da equação em função de eventuais pontos da função e, para a segunda, os pontos em função dos coeficientes.

FunctionReal

```
function FunctionReal(maindiv,coef){
FunctionReal.prototype = Object.create(FunctionObject.prototype);
FunctionReal.prototype.typeofobject=function(ty) {
FunctionReal.prototype.instantvariation = function(x){
FunctionReal.prototype.draw=function(context){
FunctionReal.prototype.closeTo=function(canvp,eixop){
FunctionReal.prototype.addDisplayDiv=function(container,editname,wcoefname) {
FunctionReal.prototype.defelmts2coef=function() {
FunctionReal.prototype.coef2defelmts=function() {
FunctionReal.prototype.moveObjTo=function(canvp,eixop,info){
FunctionReal.prototype.calculate=function(objs) {
```

A camada Controller

A função da camada *CONTROLLER* é de cuidar das interações entre o *MODEL* e o *VIEW*. Nesse sentido, existe no artefato desenvolvido duas categorias de métodos:

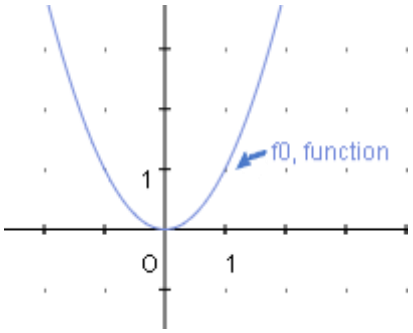
- os que vão permitir a visualização nos diversos elementos *HTML* (*CANVAS* e *DIVs*) dos objetos internos (*FunctionObject*),

- os que vão permitir ao usuário de agir sobre os objetos internos através das interfaces fornecidas pelos elementos *HTML* de representação dos objetos.

No primeiro caso dos métodos de visualização dos objetos, temos essencialmente o método *draw* do objeto *FunctionObject* encarregado de representar graficamente o objeto no *CANVAS* e simbolicamente nos outros elementos *HTML* associados a ele.

No segundo caso dos métodos de ação do usuário, temos também pertencendo à estrutura *FunctionObject*, o método *closeTo* que indica simplesmente se o curso do mouse está ou não próximo do objeto. Esse método é utilizado para permitir ao artefato dar um feedback instantâneo quando ele desloca o mouse no *CANVAS* (figura seguinte).

Feedback instantâneo



Além desse método associado à estrutura *FunctionObject*, os elementos *HTML* recebem funções gerenciando os eventos do mouse ou do teclado. Basicamente, métodos específicos são definidos e associados ao *CANVAS* pela função *addEventListener*para gerenciar os eventos *mousemove*, *mousedown* e *mouseup*, e associado aos elementos *HTML* editável com a função *onchange* para gerenciar as entradas pelo teclado.

Sobre a ferramenta Taxa de variação ()

A taxa de variação de uma função é implementada a partir de duas estruturas declinadas da estrutura *PointObj*: *PointStartTxObj* e *PointTxObj*. A primeira *PointStartTxObj* comporta-se como um ponto do grafo da função. A segunda *PointTxObj* comporta-se também como um ponto sobre o grafo da mesma função. A diferença é que ele acompanha os movimentos do ponto inicial do cálculo e, quando deslocado, define o intervalo *dx* do cálculo da taxa. O objeto *PointTxObj* na sua função *draw* representa os passos do cálculo da taxa como definidos pelo usuário.