Revista Gestão.Org, v. 16, Edição Especial, 2018. p. 279-292
ISSN 1679-1827
http://www.revista.ufpe.br/gestaoorg

# Benefícios e Limitações da Simplicidade em Projetos Inovadores de Software: uma revisão sistemática da literatura

## Benefits and Limitations of Simplicity in Innovative Software Projects: a systematic literature review

Joelson Isidro da Silva Araujo[1], Patricia Cristina Moser[1], Etelvina Raimundo Domingos[1], Vitor Abilio Sobral Dias Afonso[1], Jefferson Ferreira Barbosa[1]

[1] Centro de Informática, Universidade Federal de Pernambuco, Recife, Brasil

Correspondência: Joelson Isidro da Silva Araujo, Universidade Federal de Pernambuco, Endereço: Av. Jornalista Professor Aníbal Fernandes, S/N, Cidade Universitária, CEP.: 50.740-560 Recife, Brasil. Tel: 55 81 2126-8430. *e-mail*: jisa@cin.ufpe.br.

Recebido: 5 de outubro de 2018 Aceito: 30 de novembro de 2018 Publicado: 31 de dezembro de 2018

DOI: http://dx.doi.org/10.21714/1679-18272018v16Ed.p279-292

**Resumo**

A complexidade e incertezas fazem parte de projetos inovadores e a busca por mecanismos que amenizem tais fatores torna-se essencial para o sucesso dos projetos. Um desses mecanismos pode ser a simplicidade. Buscamos evidências sobre o uso da simplicidade em projetos de software para entender sua aplicabilidade, identificando seus benefícios e limitações por meio de uma revisão sistemática. Foram identificados 12 papers com evidências sobre diversos benefícios de sua adoção, mas poucas evidências indicaram suas limitações. Além disso, foi possível determinar que a simplicidade está mais intimamente relacionada com projetos ágeis. Esta pesquisa se mostrou restrita, uma vez que, os estudos não estão diretamente preocupados com o papel da simplicidade. O próprio ambiente inovador impõe contratempos e os benefícios identificados podem amenizar tal cenário. Entretanto, ainda há uma necessidade de mais evidências empíricas com foco no estudo da simplicidade para guiar sua aplicação de forma efetiva nos projetos.

**Palavras-chave:** Simplicidade, projetos de software, inovação.

**Abstract**

Complexity and uncertainties are part of innovative projects, so the search for mechanisms that soften these factors becomes essential for the success of projects. One of these mechanisms that can be the simplicity. We seek evidence on the use of simplicity in software projects to understand its applicability, identifying its benefits and limitations through a systematic review. 12 papers were identified with evidence about several benefits of its adoption, but little evidence indicates its limitations. In addition, it was possible to determine that simplicity is more closely related to agile projects. This research was restricted, since studies are not directly concerned with the role of simplicity. The innovative environment itself imposes setbacks and the benefits identified can soften such a scenario. However, there is still a need for more empirical evidence focusing on the study of simplicity to guide its application effectively in software projects.

**Keywords:** Simplicity, software projects, innovation.



### 1. Introdução

A complexidade do mundo moderno, oriunda da globalização e das novas variáveis de atuação das organizações, além da volatilidade e a velocidade das mudanças, impõem grandes desafios aos projetos em um contexto geral. O cenário competitivo está cada vez mais dinâmico, gerando a necessidade de inovações que, por sua vez,

impactam na estruturação e no desenvolvimento de projetos (KEELING, 2014).

Nesta perspectiva, Marinho, Moura e Maranhão (2016) afirmam que a inovação é uma das chaves para o sucesso na organização, mas que devido à incerteza e complexidade, as ameaças identificadas pela inovação são reais em projetos. Assim, enquanto a natureza de um projeto o caracteriza como um esforço temporário, por possuir início e fim definidos, com a finalidade de criar um produto ou serviço único (PMI, 2013), a inovação pode desempenhar um importante papel para um negócio, por possibilitar a competitividade no mercado com a adição de um diferencial, seja ele novo (antes não existente) ou inovador (algo existente, mas com uma nova abordagem).

Diante desses aspectos, a simplicidade pode ser um fator importante a ser trabalhada em projetos inovadores de software e, consequentemente, pode agregar valor a estes que demandam o atendimento a mudanças, novas visões e adoção de novas abordagens. Podendo assim, permitir uma maior capacidade de reação e mudança em ambientes turbulentos e inovadores, atendendo às contingências ambientais (RABECHINI; CARVALHO, 2009).

A simplicidade é uma qualidade que não só evoca a lealdade apaixonada por um design de produto, mas também se tornou uma ferramenta estratégica chave para as empresas a enfrentar suas próprias complexidades intrínsecas (MAEDA, 2006). Além disso, o desenvolvimento ágil de software demanda um foco em simplicidade, com a ideia de que a "simplicidade é essencial" (BECK; FOWLER; SCHWABER, 2001).

Dado este contexto, faz-se necessária a construção de pesquisas que trabalhem com a simplicidade em projetos de software, visando investigar seus benefícios e limitações.

Este artigo está organizado em seis seções, onde na Seção 2 é apresentado um pequeno referencial teórico com base na literatura para definir os conceitos de projetos de software, inovação e simplicidade. Na seção 3 apresentamos o método de pesquisa utilizado e todo seu processo. Os resultados estão presentes na Seção 4. Uma discussão dos resultados está presente na Seção 5 e, por fim, as conclusões, na Seção 6.

## 2. Referencial teórico

Esta seção faz um levantamento na literatura para definir os principais conceitos envolvidos nesta pesquisa, são eles, projetos de software, inovação e simplicidade.

### *2.1 Projetos de software*

O Project Management Institute (PMI), através de sua publicação mais conhecida, o guia PMBoK (PMI, 2013), define projeto como um esforço temporário com o objetivo de criar um produto, serviço ou resultado único e de natureza temporária com início, meio e fim bem definidos. O gerenciamento de projetos é a aplicação de conhecimentos, habilidades, ferramentas e técnicas às atividades dos projetos objetivando alcançar os requisitos definidos no planejamento (PMI, 2013).

De acordo com Pressman (2011), software de computador é o produto produzido e mantido por desenvolvedores de software ao longo do tempo. Esse produto abrange código-fonte, programas executáveis, documentação na forma impressa ou virtual, entre outros produtos.

Porém vários problemas podem surgir do desenvolvimento de um software que podem comprometer a qualidade do processo e consequentemente o produto final. Esses problemas foram identificados e ficaram conhecidos como a crise do software e problemas podem ocorrer em diferentes etapas do processo de desenvolvimento: análise, projeto, construção, implantação ou manutenção. Outros tipos de problemas que podem ocorrer durante o desenvolvimento de um software são especificações incorretas, sistemas desenvolvidos corretamente a partir de especificações erradas ou incompletas; e corte deliberado no escopo do projeto por limitações de prazo ou custo (SOMMERVILLE, 2007; PAULA, 2000).

Portanto pode-se concluir que projetos de software são projetos cujo produto final é o próprio software e onde são aplicadas técnicas de engenharia de software e metodologias de gerenciamento de projetos com o intuito de alcançar o sucesso do projeto.

### *2.2 Projetos de software*

A inovação do software ainda não é um termo estabelecido e as contribuições para esse campo estão espalhadas por níveis organizacionais e diferentes fases do projeto (AAEN, 2014). Ela pode ser compreendida como um processo reflexivo onde as experiências e insights durante um projeto podem mudar seu curso.

A inovação é a arte de desenvolver um novo produto, serviço ou processo baseado em uma nova ideia (BARTON, 2009), tornando-se essencial para qualquer tipo de projeto e organização, tendo como base para sua realização a

criatividade.

Nesta perspectiva, o ato de inovar consiste na necessidade de criar estratégias diferentes dos habituais meios, objetivando encontrar pequenas mudanças que tenham grandes melhorias. Em complementação, a Inovação é compreendida como a forma física da ideia de um novo produto, serviço ou processo com retorno financeiro (GORMAN, 2007), devendo-se observar a sustentabilidade e a viabilidade econômica dos projetos envolvidos.

Outro componente importante da Inovação é o refinamento contínuo e síntese de ideias, descrevendo um fluxo integrado: geração de ideias, conversão de ideias e dissolução de ideias, visualizando a Inovação como resultado final (MORRIS, 2008).

Entretanto, há mais para a Inovação do que apresentar boas ideias. Inovar envolve capitalizar bem essas boas ideias e implementá-las envolve riscos, ressaltando que tais ideias são muitas vezes filtradas em vez de totalmente realizadas (BARTON, 2009). Dado o exposto, torna-se evidente que ser inovador significa correr riscos no tocante à busca por melhorias e que, portanto, a Inovação requer cuidados e atenção especial, devendo ser, consequentemente, bem gerenciada.

Em relação a projetos inovadores de software, pode-se afirmar que o sucesso de qualquer produto envolve atingir expectativas – do cliente final, dos gerentes e dos membros da equipe (MARTINS, 2007). Dessa forma, o desenvolvimento tem se tornado cada vez mais centrado na experiência do usuário (CHEDE, 2014). Há dois aspectos particularmente importantes envolvidos no desenvolvimento de um novo produto: foco na inovação e na adaptabilidade. Produtos que não apresentem inovação, de forma a não prover valor para seus clientes, e que não podem ser facilmente adaptados às futuras necessidades, estão fadados ao fracasso (MARTINS, 2007).

*2.3 Inovação*

A simplicidade emergiu como uma chave (FLOYD; BOSSELMANN, 2013; MARGARIA; HINCHEY, 2013; MARGARIA; STEFFEN, 2010), dando a evidência que a filosofia da simplicidade é estrategicamente importante. Simplicidade é uma mentalidade, um paradigma cultural profundamente enraizado (MARGARIA; HINCHEY, 2013) e neste sentido, ela tem sido cada vez mais reconhecida como um paradigma motriz no desenvolvimento da tecnologia da informação e comunicação, manutenção, uso e gestão (FLOYD; BOSSELMANN, 2013; MARGARIA; FLOYD; STEFFEN, 2011; MARGARIA; STEFFEN, 2010; MOURA, 2011; MOURA; SKIBNIEWSKI, 2011).

No seu livro *The Laws of Simplicity*, Maeda (2006) comenta sobre o paradoxo que envolve a simplicidade e nossa revolta com a tecnologia, onde, algumas vezes, nos flagramos envolvidos até um ponto que queremos algo que seja simples e fácil de usar, mas que também faça todas as coisas complexas que gostaríamos que fosse feito. Logo, balancear simplicidade e complexidade para que se precise de menos e, de fato, se ganhe mais, não é uma tarefa fácil.

O estudo da simplicidade é uma atividade complexa e desafiadora, pois há muitas perspectivas e percepções sobre o tema e que não contribuem para uma definição única. No contexto do presente estudo, estamos particularmente interessados na análise da simplicidade em projetos inovadores de software.

Projetos de softwares tendem a ser complexos por natureza, devido a construção de suas funcionalidades que, muitas vezes, são bem peculiares dependendo de sua finalidade. Na engenharia de software e área de computação, de um modo geral, a aplicabilidade da simplicidade no desenvolvimento de projetos de software requer um estudo de conceitos para a escolha da que melhor se aplica a um determinado projeto. Tais conceitos muitas vezes podem não refletir a realidade na prática, por isso a importância de testar sua aplicabilidade em vias de fato e/ou até mesmo aperfeiçoá-la (MARGARIA; FLOYD; STEFFEN, 2011).

Um estudo realizado por Margaria, Floyd e Steffen (2011), apoiado e financiado pela União Europeia (UE), mostrou que a simplicidade em TI é "quase singular na interface humana com a tecnologia, enquanto as perspectivas de simplicidade no projeto de infra-estrutura são significativamente menos desenvolvidas". Existem várias dimensões da definição de simplicidade e suas abordagens podem ser aplicados em vários cenários de projetos de software de qualquer natureza da área da computação. Em linhas gerais, a simplicidade pode ser entendida como "uma abstração do desenvolvimento de sistemas de informação".

Revista Gestão.Org, v. 16, Edição Especial, 2018. p. 279-292
ISSN 1679-1827
http://www.revista.ufpe.br/gestaoorg

## 3. Método

Esta pesquisa foi conduzida por meio de uma revisão sistemática da literatura (RSL), uma forma de identificar, avaliar e interpretar todas as pesquisas disponíveis relevantes para uma questão de pesquisa específica, área temática, ou fenômeno de interesse.

### *3.1 Questão de pesquisa*

Conforme Gil (2010), toda pesquisa se inicia com algum tipo de problema, ou indagação e o mesmo deve ser formulado como pergunta. Para compreender o estado da arte sobre os benefícios e limitações da simplicidade em projetos inovadores de software, foi formulada a seguinte questão de pesquisa, a qual esta pesquisa visa responder:

QP: Quais os benefícios e limitações da simplicidade em projetos inovadores de software?

A resposta para a questão de pesquisa é apresentada na Seção 4.

### *3.2 Estratégia de busca*

Segundo Kitchenham *et al*. (2007) o objetivo de uma RSL é encontrar o maior número possível de estudos primários relacionados com a questão de pesquisa e para isso se faz necessário a utilização de uma estratégia de busca que deve ser determinada e seguida.

Esta pesquisa foi inicialmente conduzida por uma busca manual de publicações relacionadas à temática. Em seguida, a partir de um conjunto de 95 estudos pré-selecionados, foi definido um novo conjunto com 6 estudos, chamado de start set, para aplicação da técnica de snow balling. Chegamos a esse subconjunto a partir de uma avaliação de relevância dos estudos e buscando ter uma variedade de autores. Com a aplicação da técnica, foram obtidos 547 estudos. Por fim, foram conduzidas buscas automáticas, com aplicação da string de busca apresentada na Tabela 1, nos engenhos Scopus (325 estudos), Compendex (375 estudos), ACM (210 estudos) e IEEE (192 estudos), como um complemento à busca manual. Tanto como para a busca manual quanto para a automática consideramos o intervalo de publicações entre 2001 e 2016, partindo do princípio que a simplicidade ganhou destaque a partir do Manifesto Ágil em 2001 (BECK; FOWLER; FOWLER, 2001). Publicações de 2017 não foram incluídas, pois este foi o ano de desenvolvimento da pesquisa e estávamos interessados em buscar publicações dentro de um intervalo que compusesse anos por completo.

| |
|---|
| ("*simple*" OR "*simplicity*" OR "*simplification*" OR "*facilitate*" OR "*facilitation*" OR "*complexity*" OR "*complex*") AND ("*software project*" OR "*software development*") AND ("*innovation*" OR "*innovative*") |

Tabela 1 – String de busca.

Fonte: Autores (2017).

### *3.3 Critérios de inclusão e exclusão*

Os principais métodos empíricos aplicáveis e relevantes para a engenharia de software baseada em evidências são apresentados por Easterbrook *et al*. (2007). Sendo assim, a seleção dos estudos foi guiada pelos seguintes métodos: experimentos controlados (incluindo quasi-experimentos), estudos de caso (do tipo exploratório e de confirmação), pesquisa de opinião (*survey*), etnografias e pesquisa-ação. Bem como também, estudos com conceitos, teorias, guidelines e discussões sobre simplicidade em projetos inovadores de software.

Dessa forma, os critérios de inclusão (CI) e exclusão (CE) foram baseados na questão de pesquisa e são apresentados a seguir:

CI1: Estudos sobre a adoção da simplicidade em projetos inovadores de software;

CI2: Estudos publicados no intervalo entre 2001 a 2016;

CE1: Estudos que não apresentem dados em formato científico;

CE2: Estudos não escritos em inglês;

CE3: Resumos de palestras, tutoriais, slides de apresentação ou documentos incompletos;

CE4: Livros, teses ou dissertações;

CE5: Estudos não acessíveis;

CE6: Estudos secundários ou terciários;

CE7: Estudos apenas com lições aprendidas ou relato de experiência;

CE8: Estudos meramente baseados na opinião de especialistas

*3.4 Processo de seleção*

A pré-seleção inicialmente foi guiada por uma busca manual simples feita, de forma individual, pelos pesquisadores. Ao unir os resultados, foi gerada uma lista com 95 estudos e, destes, foram selecionados 6 para composição do *start set*, para o processo de *snow balling*. Com a aplicação da técnica de *snow balling* foram obtidos 547 estudos, identificados em dois ciclos na fase "para trás":

1º ciclo: referências do conjunto de estudos do *start set*;

2º ciclo: referências dos estudos incluídos no 1º ciclo;

E, um ciclo na fase "para frente":

1º ciclo: citações do conjunto de todos estudos previamente incluídos na fase "para trás".

Desse modo, é possível perceber que o *snow balling* não foi conduzido até seu esgotamento e isto se deve ao próprio contexto explorado, pois além de não existirem muitos estudos focados em simplicidade, na medida que aumentávamos nossa amostra, percebemos que não estávamos agregando novas descobertas. Nesse sentido, foi feita uma adaptação para recorte da pesquisa, que reflete também no número final de estudos a serem sintetizados.

Todo o processo de decisão sobre inclusão e exclusão foi feito em par para cada estudo analisado e, para garantir isto, o grupo de pesquisa foi subdividido em 3 duplas. Possíveis conflitos entre uma dupla foram resolvidos alternativamente por um membro de outra dupla. Todo processo de seleção é ilustrado na Figura 1.

Na busca automática, inicialmente foram obtidos 1.102 estudos, mas destes, 195 eram duplicados e foram removidos, restando 907.

Os estudos da busca automática foram agrupados com os da busca manual. Com isso, novas duplicações foram encontradas e, após a aplicação dos critérios de inclusão, restaram apenas 81 estudos potencialmente relevantes. Após aplicação dos critérios de exclusão, 13 estudos foram removidos, restando 68. Aplicamos um novo refinamento, fazendo um filtro com a avaliação de todos os pesquisadores para cada um dos estudos. Cada pesquisador fez a leitura dos estudos e deu uma nota que poderia ser 0 ou 1, de modo que, a nota máxima para aquele determinado estudo, deveria ser 6. Foram mantidos os estudos com nota 5 e 6 e, com isso, foram removidos mais 56 estudos, restando os 12 finais.

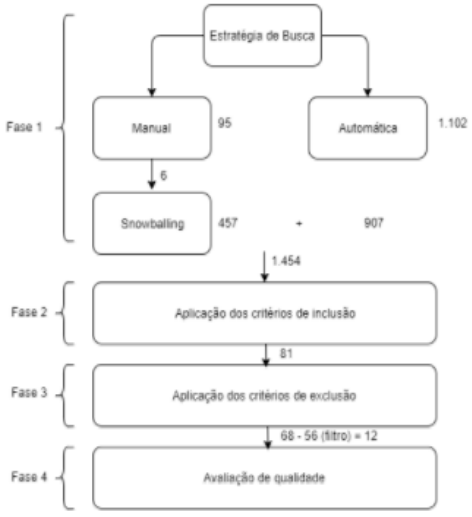


Figura 1 – Estratégia de busca.
Fonte: Autores (2017).

Revista Gestão.Org, v. 16, Edição Especial, 2018. p. 279-292
ISSN 1679-1827
http://www.revista.ufpe.br/gestaoorg

*3.5 Extração de dados*

A extração de dados foi feita em uma planilha Microsoft Excel®, onde foram definidas algumas informações que deveriam ser extraídas dos estudos. Essas informações podem ser vistas na Tabela 2.

| | |
|---|---|
| **ID** | Identificador do estudo |
| **Título** | Título do estudo |
| **Ano** | Ano de publicação |
| **Contexto** | Contexto do estudo |
| **Problema** | Problemática abordada no estudo |
| **Categoria** | Classificação do tipo de simplicidade |
| **Benefícios** | Benefícios da simplicidade |
| **Limitações** | Limitações da simplicidade |
| **Inovação** | Cenário inovador |
| **Considerações** | Considerações que enfatizam uma evidência |

Tabela 2 – Tabela de extração dos dados.
Fonte: Autores (2017).

*3.6 Síntese dos dados*

Nesta revisão foi utilizada uma abordagem indutiva para sintetizar os dados onde, a partir de dados particulares, suficientemente constatados, é inferida uma verdade geral ou universal, não contida nas partes examinadas, com o objetivo de levar a conclusões com conteúdo mais amplo do que as premissas nas quais o argumento criado se baseia (LAKATOS, 2011). Dada essa abordagem, utilizamos uma técnica de síntese qualitativa, a síntese de linha de argumento (KITCHENHAM, 2007).

## 4. Resultados

Foram analisados 12 estudos, publicados entre 2001 e 2016. A Tabela 3 sumariza esses estudos, ordenados por um identificador (ID), seguido de informação sobre autor(es) e ano de publicação.

A partir destes estudos, buscamos responder nossa questão de pesquisa: "Quais os benefícios e limitações da simplicidade em projetos inovadores de software?". Buscar respostas para esta questão não é uma tarefa simples devido a subjetividade que pode ser dada ao conceito simplicidade, bem como por não ser um termo estabelecido na área (SANTOS, 2016). O mesmo acontece com o conceito de inovação (AAEN, 2014). Dessa forma, alinhamos internamente na equipe qual seria nossa posição sobre ambos os conceitos, para que soubéssemos selecionar estudos que atendessem nosso interesse, cientes de que poderíamos deixar de identificar estudos possivelmente relevantes, mas seguros de que esse seria nosso processo adotado para abordar o problema.

Sendo assim, consideramos simplicidade como sendo qualquer forma de prover a redução da complexidade dos projetos. E inovação como sendo a própria natureza dos projetos que os caracterizam como únicos (temporário, com início e fim definidos com o desenvolvimento de um produto ou serviço específico).

| **ID** | **Autor** | **Ano** |
|---|---|---|
| S0031 | L. Sha | 2001 |
| S0050 | Santos, Wylliams | 2016 |
| S0083 | T. Margaria, B. D. Floyd, and B. Steffen | 2011 |

| | | |
|---|---|---|
| S0127 | Aaen, Ivan | 2013 |
| S0166 | Harkema, S.J.M., Baets, W. | 2001 |
| S0410 | Aaen I., Jensen R.H. | 2014 |
| S0431 | Trier, Matthias and Richter, Alexande | 2013 |
| S0436 | Sam, Shalu Achamma et. Al | 2016 |
| S0613 | Bosch J. | 2013 |
| S0674 | Bonner N.A., Teng J.T.C., Nerur S. | 2010 |
| S1071 | Li, Zheng; O'Brien, Liam; Yang, Ye | 2014 |
| S1417 | Barton B. | 2009 |

Tabela 3 – Papers selecionados.

Fonte: Autores (2017).

Dentro dos estudos alguns fatores foram identificados e classificados como simplicidade em projetos de software, entre eles, a prototipação, a facilidade de uso, a evolução contínua e o desenvolvimento incremental, o reuso, a decomposição. A Tabela 4 sumariza esses fatores classificando-os em benefícios e limitações que respondem nossa questão de pesquisa. Para elaborá-la, os benefícios e limitações presentes em cada artigo foram identificados, categorizados e posteriormente reduzidos a uma única expressão que refletisse o real sentido do benefício e/ou da limitação.

A seguir, apresentamos nossa síntese, construída com argumentos derivados da extração dos dados. Os argumentos são seguidos de trechos extraídos do estudo fonte.

A adoção de metodologias ágeis no desenvolvimento causa uma percepção de menos complexidade para os desenvolvedores. Desse modo, adotar a agilidade nos projetos acaba sendo um benefício que traz consigo o ideal da simplicidade [S0674]: "*agile development methods lead to developers' beliefs that they are less complex, more compatible and provide increased benefits*".

Ainda nesta perspectiva, a metodologia ágil Scrum, permite a implementação de métodos enxutos para as organizações, fornecendo ferramentas simples, eficazes e técnicas que se estendem além do software. Quando Scrum se estende para a Organização, a inovação sob a forma de criação de valor é um subproduto natural. O Scrum foca em entregar o maior valor comercial no menor tempo, possibilitando agilidade ao processo, o que simplifica o desenvolvimento [S1417]: "*All-out Organizational Scrum provides an innovation value chain by implementing transparency across the organization, managing the idea process, using short Sprint cycles to evaluate the output of the cycles, encouraging frequent releases (diffusion) and removing obstacles in priority order*".

O desenvolvimento ágil de software exige um foco na simplicidade, por entender que esse componente é essencial e importante para o sucesso de um projeto. Como a redução, o tempo, a estrutura, a organização, a decomposição, entre outras características, pode ser considerados elementos da simplicidade, o desenvolvimento ágil deverá se beneficiar com sua adoção [S0050]: "*it is important to develop an approach to support the agile team in order to foster simplicity in Agile Software Development*".

Outro fator associado à simplicidade, descoberto nesta pesquisa, é a prototipação. Ela se dá por meio da construção de protótipos, auxiliando no processo de descobertas e refinamento de requisitos, bem como na identificação de novas opções e desafios para o desenvolvimento de software. A simplificação por meio de protótipos permite, também, a observação de novas características que podem tornar o projeto mais rico, além de ser uma forma de melhorar o escopo visualmente [S0127]: "*a prototype corresponds to a perceived use context and a project vision, and it consists of a configuration of features and platforms. This can be evaluated in connection with a sprint review meeting, and context, vision, and configuration may change if the team chooses to search for more valuable*

*solutions*".

A prototipação também beneficia o desenvolvimento gradual do software, uma vez que os protótipos ajudam a entender problemas e podem melhorar problemas de comunicação [S0410]: "*The prototype can be evaluated in connection with a sprint review meeting, and scope, needs, vision, and configuration may change… helped inspire an answer to the communication problems*" e "*establish an end-in-view allowing the team to work determined under uncertainty*".

Por outro lado, o excesso de protótipos exerce um impacto negativo à simplicidade. Isso porque, esse excesso pode causar estresse ocupacional devido ao esforço despendido [S0127]: "*A weakness might be an increased risk of occupational stress*".

Foi possível identificar que existe uma relação entre a complexidade de um produto de software e o esforço despendido para seu desenvolvimento, onde se a complexidade aumenta enquanto o esforço real diminui, os projetos contam com programadores com capacidades superiores ao tamanho do produto. Sendo assim, podemos concluir que projetos mais simples, demandam um menor esforço, embora seja buscado desenvolver soluções complexas com menos esforço. Por outro lado, projetos complexos dificultam o desenvolvimento e, consequentemente, o entendimento do produto desenvolvido [S1071]:

- "*For software projects, when product complexity increases, the actual effort can still decrease due to the increased capabilities of programmers*";
- "*For software projects, when product complexity increases, the actual effort can still decrease due to the decreased product size or/and database size*";
- "*it is impossible to infinitely increase product complexity without increasing actual effort by adjusting other factors. Therefore, it is worth investigating further to what extent people can spend less effort for a more complex software product*";
- "*it is possible to construct an extremely complex and large software product by developing and composing smaller modules*";
- "*the reduced product scale can also decrease actual effort even when product complexity increases*";
- "*the complexity in a software product incurs difficulty for people to understand the development process and thus the product itself*".

Podemos dizer que a capacidade humana dita o esforço despendido. Se essa capacidade é alta, não haverá grandes problemas de entendimento, nem de esforço que poderia ser despendido excessivamente. Mas se essa capacidade é baixa, o simples pode ser difícil de entender, exigindo um esforço grande que pode ser desnecessário, por não gerar resultados positivos [S1071]: "*Human capability (in terms of the capabilities of programmers and/or analysts) and product scale (in terms of the size of product and/or database) are two main factors for the negative correlation between actual effort and product complexity in software developments*".

Analisando a simplicidade a partir da complexidade, podemos dizer que a redução da complexidade possibilita simplificar a evolução dos produtos. Essa simplificação por ser alcançada pela divisão do software em camadas, desacoplando funcionalidades em outra camada, tornando o desenvolvimento mais eficiente e ágil [S0613], ou ainda por uma análise de quebra da complexidade [S0436]. Com isso, podemos dizer que pode haver uma redução de erros, bem como pode ser mais fácil projetar, construir e usar a solução [S0436]:

- "*Software developers a potent solution by separating commoditized, differentiating, and innovation and experimentation functionality into three layers*";
- "*Faced with the implications of unmanaged complexity, companies need alternative approaches to help maintain the elegance and simplicity of product architecture as it evolves*";
- "*The task analysis is to break a single complex code into different ones that can be later pooled*";
- "*Complexity of a program will increase the number of errors in that particular program. If the complexity of a system is higher than expected, then it is more difficult to design, build, and use".*

Um sistema de software tende a ser mais complexo se os requisitos tiverem mais funcionalidades. Essa complexidade é desenvolvida com base em análise e síntese de tarefas que, quando bem parametrizadas, permitem a quebra do processo maior e complexo em menores partes e mais simples. Essa simplicidade pode ser alcançada pela diluição da abstração da solução e até mesmo por meio do reuso [S0436]:

- "*The software development process can be easier and the resultant software will be robust, simple and maintainable. As a result software can be reused and the underlying cost involved in maintenance can be reduced*";

- "*Complexity in software development can be based on task analysis and synthesis. Task analysis is to break a single complex code into distinct ones which can be later brought together. This is done especially in the case of existing programs. Synthesis involves constructing an artefact to satisfy the requirements. For a simple program, the requirements are always simple and hence synthesis can proceed directly*".

Neste aspecto, existem limitações impostas pela complexidade que são identificadas na fase da construção de artefatos para atender os requisitos desse tipo de sistema, pois nele a complexidade aumenta o número de erros, tornando mais difícil projetar, construir e usar a solução. Essas limitações ainda são intensificadas quando o fator humano (desenvolvedor ou engenheiro) tem dificuldade para compreender a solução. Podemos concluir assim, que o uso da simplicidade possivelmente pode minimizar esses efeitos ocasionados pela complexidade [S0436]: "*Complexity of a program will increase the number of errors in that particular program. If the complexity of a system is higher than expected, then it is more difficult to design, build, and use. The success completely depends on human understanding in the software development*"

Há um desafio em melhorar a confiabilidade e disponibilidade do software que se torna mais complexo, neste sentido, podemos dizer que a melhoria pode ser alcançada com a introdução da simplicidade. Entretanto o usuário não se satisfaz com o simples, pois esperam algo complexo [S0031]: "*the existence of a simple and reliable component can be used to guarantee the critical properties*" e "*users are not satisfied with simple software. They want functionalities and features provided by complex software. Fortunately, this dilemma can be addressed by "using simplicity to control complexity*".

A redução de elementos faz com que os usuários se familiarizem com funcionalidades passo a passo. Eles reconhecem como isso enfatiza a parcimônia e a elegância [S0431]: "*Simplicity in design is achieved by reducing to the essential functional elements and principles (ontological simplicity, i.e. parsimony) as well as by emphasizing simple organization of structures and layouts of the functional elements*".

A transparência percebida pelos usuários, em função da simplicidade da interface, leva a um melhor controle do software percebido pelos usuários. Além disso, a simplicidade e a satisfação de uso estão muito conectadas, pois os usuários percebem a redução da complexidade bem como uma melhor experiência de uso do software, ficando mais à vontade com o uso de uma plataforma. Porém, a simplicidade percebida é relativa, ela pode ser influenciada de forma positiva através de efeitos da rotina de trabalho, como por exemplo, a adição incremental de elementos ao longo do tempo ou então pelo costume pelo uso do artefato. O usuário pode achar simples, mas na verdade, pode estar familiarizado ao software. Podemos dizer então, que a simplicidade do projeto está positivamente relacionada à melhor compreensão do software pelo usuário [S0431]:

- "*Perceived simplicity is linked with perceived control*";
- "*Increased simplicity is linked with perceived ease of use*";
- "*Perceived simplicity is relative: It is positively influenced by routinization effects like (a) the incremental addition of elements over time and (b) the relative familiarity of the artifact*";
- "*Simplicity of the design is positively related to software adoption success*".

A simplicidade dos sistemas está relacionada a um alto grau de maleabilidade e baixa prescritibilidade. Em contraste com a maioria dos softwares clássicos que são orientados a casos de uso e criados para suportar um processo específico, alguns softwares suportam padrões funcionais muito básicos e abstratos (por exemplo, publicação de fragmentos de informações em Microblogging, edição e adição de texto em *wikis*). Eles podem ser facilmente compreendidos pelos usuários ou serem aplicados a um uso individual em uma grande variedade de situações. Porém, essa maleabilidade requer estratégias especiais de implementação para conseguir uma apropriação coerente e fiel do sistema e obter benefícios organizacionais. É importante incorporar estas plataformas com as práticas de trabalho existentes na organização e enfatizar seu uso. Isso deve acontecer iterativamente durante um longo período de tempo, a fim de manter a percepção da simplicidade [S0431]: "*Simplicity is related to a high degree of malleability*" e "*Malleability requires special implementation strategies in order achieve a coherent and faithful appropriation of the system and to derive organizational benefits*".

A simplicidade pode, então, melhorar a qualidade dos sistemas, mantendo as coisas o mais simples possível. O uso da simplicidade pode deixar os sistemas previsíveis e transparentes, e as coisas se tornam mais simples e mais fáceis de usar quanto mais as entendemos [S0083]:

- "*where the results of applying these principles improves system quality by keeping things as simple as possible*";

- "*our participants characterized simplicity along several other dimensions such as 'no surprises', predictable, and transparente*";
- "*Simplicity is connected to making things understandable with as little knowledge as possible about that particular thing*";
- "*Simplicity is reducing something into its essential complexity and getting rid of all the accidental complexity given the specific purpose of the system under consideration*";
- "*Simplicity is not an innate property of something but it depends on the purpose of the thing you want to use*";
- "*Such a purpose is important; the essence of simplicity is from the perspective of the 'user', be it an end user or a system developer. Things become simpler the more we understand them*".

Ao estabelecer uma relação entre a simplicidade e a inovação, se não houver um bom alinhamento e controle no processo de desenvolvimento, incertezas e imprevisibilidade podem surgir. Nesse contexto, a inovação pode ser vista como um sistema complexo por causa das relações entre os envolvidos e pela má compreensão simplista que pode não ser suficiente para explicar algo em um nível mais geral [S0166]:

- "*Not so much from the point of view of the interaction taking place or the relational point of view; but from the perspective of decision making*";
- "*In essence the setup is based on a notion that actions of individuals have to be regulated and the process controlled as much as possible in order to reduce uncertainty and unpredictability*";
- "*Innovation is then a complex system because of the intricate relations between its members and because understanding of the behavior of the parts does not suffice to explain what is happening at a more general level*".

Ainda assim, a simplicidade pode ser introduzida em projetos inovadores, tendo a visão de que a inovação é um "sistema aberto" e flexível, condicionada por fatores que podem ser controlados e influenciados, em parte, sendo assim previsíveis. Logo, os resultados não estão condicionados apenas aos aspectos tangíveis, como o produto, mas também pelo grau em que o conhecimento é compartilhado, trocado e adquirido pelos membros, gerando maturidade e conhecimento [S0166]: "*Innovation is regarded as an "open system" constrained by factors that can be controlled, influenced and, in part, be predicted. The driving motor is the interaction between organizational members and other stakeholders who have something to gain from the product innovation. The outcome will in part measured by tangible aspects, like the physical product. It will however also be measured by the extent to which knowledge is shared, exchanged and gained by the network members*".

| **Benefícios** | **Limitações** |
|---|---|
| Redução de elementos [S0431] [S0083] | Satisfação dos usuários [S0031] |
| Facilidade de uso [S0431] [S0083] | Desconhecimento em desenvolvimento [S0436] |
| Maleabilidade [S0431] | Imprevisibilidade no desenvolvimento [S0166] |
| Evolução contínua [S0127] [S0410] | Subestimar a simplicidade [S0127] |
| Redução de erros [S0436] | |
| Melhor compreensão [S0436] [S0431] | |
| Menos esforço [S1071] | |
| Facilitar a adoção e implantação de sistemas [S0613][S0436] | |
| Confiabilidade [S0031] | |
| Redução da complexidade [S0613] [S0674] | |

| | |
|---|---|
| Redução de tempo [S1417] [S0050] | |
| Maior controle [S0431] | |
| Agilidade [S1417] [S0674] | |

Tabela 4 – Benefícios e limitações da simplicidade.

Fonte: Autores (2017).

Podemos concluir nossa resposta à pergunta de pesquisa, da seguinte forma. A simplicidade parece estar mais associada a projetos ágeis e um dos primeiros benefícios atingidos pelo o uso da simplicidade é a redução de elementos. Essa redução pode ser desde uma interface simples, ou mesmo, a remoção de funcionalidades que não são essenciais para um sistema. Podemos obter em resposta a essa redução, uma melhor facilidade de entendimento, que está diretamente relacionado à facilidade de uso.

Uma das formas para compreender a simplicidade é percebê-la a partir da redução da complexidade. Neste aspecto, é possível despender menos esforço para o desenvolvimento de uma dada solução, o que impactará na redução de tempo. Podemos inferir, ainda, que o desenvolvimento de algo menos complexo nos permite um melhor controle sobre todo o processo relacionado ao seu desenvolvimento e, consequentemente, produz uma solução menos propícia a erros.

A prototipação também pode ser compreendida como uma forma de simplicidade. Ela auxilia o processo da evolução contínua e pode facilitar a compreensão de entendimento do problema, bem como de requisitos, além de poder facilitar a comunicação entre a organização e o cliente. Com o uso de um processo de desenvolvimento contínuo obtemos maleabilidade, conseguindo flexibilidade ao ponto de ser capaz de se adaptar de forma rápida a possíveis mudanças.

Diante dos benefícios proporcionados pelo uso da simplicidade, fica também evidente a possível facilidade de adoção e implantação de sistemas que adotam uma abordagem simplista. Se mesmo simples, uma solução é capaz de atender as necessidades essenciais de um dado problema, podemos dizer que essa solução obteve êxito. Esse êxito está associado a resultados positivos da solução com geração de valor para o cliente, que passa a ter uma maior confiabilidade no que foi entregue.

Ao avaliar a contrapartida do uso da simplicidade podemos identificar a satisfação do usuário como uma limitação, isso porque o simples pode não atender às expectativas do usuário, que espera algo mais complexo. Da mesma forma, podemos dizer que o uso excessivo da simplicidade pode provocar confusão e tornar uma dada solução mais complexa de se entender.

A capacidade cognitiva humana, quando voltada para desenvolvimento de produtos, também pode ser compreendida como uma limitação, visto que, a simplicidade, por si só, não capacita profissionais de desenvolvimento de software a desenvolverem produtos que sejam simples, porém funcionais. Ou seja, profissionais que não têm o conhecimento sobre programação, não conseguirão programar pelo fato de o produto ser simples.

Por fim, podemos dizer que a adoção da simplicidade pode auxiliar o desenvolvimento de projetos inovadores, embora tenhamos que lidar com o fato de que o próprio ambiente inovador impõe certa imprevisibilidade.

## 5. Discussões

No início desta pesquisa imaginamos obter mais estudos que respaldassem o trabalho, ou seja, esperávamos encontrar mais trabalhos abordando a simplicidade como um dos fatores essenciais para projetos inovadores de software. Contudo, diante dos nossos resultados, fica evidente certa dificuldade em encontrar evidências suficientes para apoiar nossa resposta à seguinte pergunta de pesquisa: “Quais os benefícios e limitações da simplicidade em projetos inovadores de software?”. Da mesma forma, não encontramos até a finalização dessa revisão sistemática trabalhos relacionados que tivessem preocupados com a adoção da simplicidade em projetos de software, para que assim, pudéssemos fazer um comparativo. Encaramos o desafio de trabalhar com a simplicidade relacionada a projetos inovadores de software, mesmo conhecendo a dificuldade existente em definir o que é simplicidade, dado que a literatura não traz uma definição concreta sobre. Assim, nos preocupamos em

apresentar mesmo que de forma inicial conceitos que auxiliem o seu entendimento e aplicabilidade por meio da identificação de seus benefícios e limitações. Obtemos alguns resultados que embasaram nossa síntese e podemos inferir sobre três aspectos: *i*. existem poucos estudos com foco em simplicidade; *ii*. existem mais estudos abordando benefícios do que limitações e *iii*. a simplicidade está associada ao desenvolvimento ágil.

Percebe-se a necessidade de pesquisas que se concentrem no estudo do uso da simplicidade em projetos de software, pois seus benefícios são promissores para adoção em projetos inovadores que enfrentam certa turbulência pelo "novo", estando mais propícios ao insucesso ao enfrentar complexidades. Nesse contexto, também foi difícil identificar possíveis limitações sobre o uso da simplicidade, o que pode ser mais uma evidência da necessidade de pesquisa com foco na temática que avaliem mais a fundo fatores associados a simplicidade, identificando qual o impacto deles sobre os projetos de software. Deste modo, possíveis ameaças causadas pela simplicidade podem ser conhecidas e mitigadas.

Apesar de a simplicidade ainda ser um conceito subjetivo, de difícil compreensão, já existe pesquisa buscando estabelecer o conceito dentro do contexto de projetos de software (SANTOS, 2016). Percebe-se, ainda, que ela está mais associada a projetos ágeis, e isso é explicado pelo próprio Manifesto Ágil, que defende a simplicidade como sendo essencial para o sucesso dos projetos (BECK; FOWLER; SCHWABER, 2001).

Dentro do que foi analisado por este estudo, podemos identificar alguns benefícios e limitações, observados na tabela 4. Alguns são mais evidentes, ou seja, parecem mais claros, enquanto que outros não são facilmente compreendidos e demandam uma maior apropriação para compreensão e interpretação. Mas, o mais importante foi apresentá-los como uma forma de contribuir com o conhecimento sobre a simplicidade em projetos de software e, em consequência, poder transferir esse conhecimento para que a comunidade acadêmica desperte seu interesse pela temática para que, então assim, possam produzir resultados para a prática.

## 6. Conclusões

É crescente a preocupação das empresas em gerar valor a partir de novos produtos e serviços, com o objetivo de aumentar a competitividade e manter um ritmo sustentável de crescimento. Esse contexto contribui para um ambiente propício à inovação, que vai desde a geração de ideias e criação de estratégias até o desenvolvimento de novos produtos. Sabe-se que a inovação é uma das chaves para o sucesso na organização, por outro lado, devido à incerteza e complexidade, as ameaças identificadas pela inovação são reais em projetos de software.

Dentro desse cenário surge a simplicidade como um importante fator a ser trabalhado para permitir a geração de um ambiente favorável à inovação e apto a lidar com os contratempos.

De acordo com o objetivo deste estudo, "Identificar evidências sobre o uso da simplicidade em projetos inovadores de software para entender sua aplicabilidade, identificando seus benefícios e limitações", foi realizada uma revisão sistemática da literatura e foram identificadas e colhidas evidências de que o uso da simplicidade pode trazer inúmeros benefícios, como a redução de elementos, a melhor compreensão, a facilidade de uso, a redução de tempo, dentre outros. Por encontramos algumas poucas evidências sobre limitações da simplicidade, acreditamos que sua utilização em projetos de software se mostra mais benéfica, embora isso não seja uma verdade absoluta. Cabe, então, uma investigação maior sobre os possíveis impactos negativos.

Assim, esperamos com este estudo evidenciar a necessidade de novas pesquisas na área com foco em simplicidade e, dessa forma, despertar o interesse dos pesquisadores pela temática que se mostra promissora para redução da complexidade dos projetos de software, bem como, para auxiliar no processo da tomada de decisão.

## Referências

AAEN, I.; JENSEN, R. H.; (2041) Pragmatic software innovation. In: **International Working Conference on Transfer and Diffusion of IT**. Springer, Berlin, Heidelberg, p. 133-149.

BARTON, B.; (2009) All-Out Organizational Scrum as an Innovation Value Chain. In **Proceedings of the 42nd International Conference on System Science**, 2009.

BECK, K.; FOWLER, M.; SCHWABER, K. (2001) Manifesto for Agile Software Development, 2001. Disponível em: http://agilemanifesto.org.

CHEDE, C.; (2014) O desafio de desenvolver sistemas inovadores cada vez mais rápido. **IBM Community**,

Disponível em: <https://www.ibm.com/developerworks/community/blogs/ctaurion/entry/o_desafio_de_desenvolver_sistemas_inovadores_cada_vez_mais_rapido>.

EASTERBROOK, S. M.; et al. (2007). Selecting Empirical Methods for Software Engineering Research. In F. Shull, J. Singer and D. Sjøberg (eds) Guide to Advanced Empirical Software Engineering, **Springer**.

FLOYD B. D.; BOSSELMANN. S.; (2013) ITSy - Simplicity Research in Information and Communication Technology. **Computer**, 46(11), pp.26–32, Nov.

GIL, A. C. (2010) Como elaborar projetos de pesquisa. 5. ed. - São Paulo: **Atlas**.

GORMAN, T. (2007) Innovation, F+W Publications, Avon, MA.

KEELING, R. (2014) Gestão de projetos – Uma Abordagem Global. 3. ed. Rio de Janeiro: **Saraiva**.

KITCHENHAM, B.; CHARTERS, S. (2007) Guidelines for Performing Systematic Literature Reviews in Software Engineering, **Keele University**. [S.I.].

LAKATOS, E. M.; MARCONI, M.; (2011) A. Fundamentos de Metodologia Científica. São Paulo: **Atlas**.

MAEDA, J.; (2006) The laws of simplicity. **MIT press**.

MARANHÃO, R.; MARINHO M.; MOURA H.; (2016) Directions for Building an approach to innovate software project management, In: **13th International Conference on information system & Tecnology Management** – CONTECSI.

MARGARIA, T.; STEFFEN, B.; (2010). Simplicity as a Driver for Agile Innovation. **Computer**, 43(6):90–92, June.

MARGARIA, T.; HINCHEY, M.; (2013). Simplicity in IT: The Power of Less. **Computer**, 46(11):23–25, Nov.

MARGARIA, T.; FLOYD, B. D.; STEFFEN, B.; (2011). IT Simply Works: Simplicity and Embedded Systems Design. In **Computer Software and Applications Conference Workshops** (COMPSACW), 2011 IEEE 35th Annual, pages 194–199. IEEE, July.

MARTINS, J. C. C.; (2007) Técnicas para Gerenciamento de Projetos de Software. Rio de Janeiro: **Brasport**.

MORRIS, R.; (2008) From acorns of ideas to oak trees of business success, http://www.amazon.com/review/R1V6GI73D4EDX.

MOURA, H.; SKIBNIEWSKI, M.; (2011) The Evolution of Management Thinking. In: **International Research Network on Organizing by Project** (IRNOP).

MOURA, H.; (2011) Software Project Framework. **Technical report**, Federal Univerity of Pernambuco, Recife, Pernambuco, 2011. Disponível em: http://www.cin.ufpe.br/hermano/spf/SPF%20The%20Framework%20br%20v1.pdf.

PAULA FILHO, W.; (2000) Engenharia de Software: Fundamentos, Métodos e Padrões. 2ª Ed., Rio de Janeiro: **LTC**.

PRESSMAN, R.; (2011) Engenharia de Software: Uma Abordagem Profissional. 7ª Ed., São Paulo: **McGraw-Hill**.

PMI; (2013) A Guide to the Project Management Body of Knowledge: PMBOK® Guide, 5th edition - EUA, Project Management Institute.

RABECHINI JR., R.; CARVALHO, M. M.; (2009). Gestão de Projetos Inovadores em uma Perspectiva Contingencial: análise teórico-conceitual e proposição de um Modelo, In **Revista de Administração e Inovação**. São Paulo, v. 6, n. 3 (pp. 63-78).

SANTOS, W.; (2016). Towards a better understanding of simplicity in Agile software development projects. In: **Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering**. ACM, 2016. p. 2.

SOMMERVILLE, I.; (2007). Engenharia de Software. 8ª Ed., São Paulo: **Addison Wesley Brasil**.

**Referências dos estudos incluídos**

[S0031] L. Sha. Using Simplicity to Control Complexity. IEEE Software, 2001.

[S0050] SANTOS, W. Towards a Better Understanding of Simplicity in Agile Software Development Projects. Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering, 2016.

[S0083] MARGARIA, T. FLOYD, BD. STEFFEN, B. IT Simply Works: Simplicity and embedded systems design. Computer Software and Applications Conference Workshops, 2011.

[S0127] AAEN, I. Software Innovation – Values for a Methodology. Scandinavian Conference on Information Systems, 2013.

[S0166] HARKEMA, S.J.M, BAETS, W. "Customerized" innovation through the emergence of a mutually adaptive and learning environment. European Journal of Economic and Social Systems, 2001.

[S0410] AAEN, I., JENSEN R. H. Pragmatic Software Innovation. International Working Conference on Transfer and Diffusion of IT, 2014.

[S0431] TRIER, M., RICHTER, A. " I Can Simply..."-Theorizing Simplicity As A Design Principle And Usage Factor. Proceedings of the 21st European Conference on Information Systems, 2013.

[S0436] SAM, S. A., THANUSHA, M., MANJULA, R. A STUDY ON COMPLEXITY IN PROGRAMS. International Journal of Pharmacy & Technology, 2016.

[S0613] BOSCH, J. Achieving Simplicity with the Three Layer Product Model. Computer IEEE, 2013.

[S0674] BONNER, N.A., TENG. J.T.C., NERUR, S.P. The Perceived Advantage of Agile Development Methodologies by Software Professionals: Testing an Innovation-Theoretic Model.AMCIS, 2010.

[S1071] LI, Z., O'BRIEN, L., YANG, Y. Impact of product complexity on actual effort in software developments: An empirical investigation. Software Engineering Conference, 2014.

[S1417] BARTON, B. All-Out Organizational Scrum as an Innovation Value Chain. System Sciences, 2009.