



Departamento de Ciências e Tecnologias de Informação

Animation of BPMN Business Processes Models

Marco André Marques Roque

A dissertation presented in partial fulfillment of the requirements for the degree of
Master in Computer Science and Business Management

Supervisor:
Fernando Brito e Abreu, Associate Professor,
ISCTE-IUL

July, 2014

[This page is intentionally blank]

Agradecimentos

À minha família, em especial aos meus pais Ana Roque e António Roque, pela força e apoio incondicional que sempre prestaram durante a minha vida.

À minha namorada Vanessa Matos, pela paciência, força e ajuda.

A todos os meus amigos que me incentivaram a concluir este desafio, especialmente ao André Tomás pelo seu apoio durante a elaboração da dissertação.

Ao Professor Doutor Fernando Brito e Abreu, pelos ensinamentos na arte de fazer Ciência, pelo seu tempo e dedicação que me permitiram alcançar os meus objetivos académicos.

[This page is intentionally blank]

Abstract

BPM in the business world is currently supported by tools that facilitate the design, implementation, execution, monitoring and optimization of business processes. These so called Business Process Management Suites usually have animation capabilities associated to process simulation. However, animation capabilities vary depending on the tool and the better these are, the higher is the animation preparation effort. This problem is more evident when generic simulation tools are compared with BPM specific ones, which use BPMN (Business Process Modeling Notation) as the modeling notation and have more limited animation capabilities.

This dissertation presents a proposal to endow the BPMN with animation capabilities which respects all the elements presentation rules established in the notation specification. This proposal was designed based upon the data collected through the application of the animation capabilities evaluation taxonomy also proposed here in this dissertation. A prototype was built on top of an open-source tool in order to implement our animation proposal and was used to animate the service request process model using real execution data from an IT Service management tool used at ISCTE-IUL.

Keywords:

BPMN models animation; business process models animation; process execution data replay.

[This page is intentionally blank]

Resumo

A gestão de processos de negócio no mundo empresarial é actualmente suportada por ferramentas computacionais, que facilitam o seu desenho, implementação, execução, monitorização e optimização. Essas ferramentas, vulgarmente designadas de *Business Process Management Suites*, possuem usualmente mecanismos de animação aliados à simulação de processos. Contudo, as capacidades de animação diferem consoante a ferramenta, sendo que quanto melhores são estas capacidades, maior é o tempo investido na preparação da animação. Este problema torna-se mais evidente quando são comparadas ferramentas de simulação mais genéricas com ferramentas dedicadas à gestão de processos de negócio, que usam BPMN (*Business Process Modeling Notation*) como notação de modelação, em que as animações se tornam mais limitadas.

Esta dissertação apresenta uma proposta de animação para a notação *BPMN* que respeita as regras de apresentação dos elementos da notação estabelecidas na especificação da mesma. Esta proposta foi desenhada com base nos resultados recolhidos através da aplicação de uma taxonomia também aqui proposta para a avaliação das capacidades de animação de ferramentas de simulação de processos de negócio, onde se reflecte o estado da arte no campo da animação de processos. A proposta de animação foi implementada num protótipo, que assenta sobre uma ferramenta *open-source* seleccionada a partir de requisitos definidos e apresentados na dissertação. Por fim, o protótipo foi usado para animar um modelo do processo de requisição de serviços usando, para isso, dados de execução reais recolhidos da ferramenta de gestão de serviços de TI utilizado no ISCTE-IUL.

Palavras – chave:

Animação de modelos BPMN; reprodução de processos; animação de modelos de processos de negócio.

[This page is intentionally blank]

Table of Contents

1. Introduction	2
1.1. Motivation	2
1.2. Research problem	2
1.2.1. Problem statement	2
1.2.2. Research questions	2
1.3. Expected contributions and objectives	3
1.4. Research methodology	4
1.5. Business process models animation	9
1.6. BPMN – Business Process Modeling Notation	10
1.7. Document structure	11
2. A taxonomy on tool support for process animation	14
2.1. Business Process Simulation	14
2.1.1. BPS benefits	14
2.1.2. BPS limitations	15
2.2. The taxonomy categories	16
2.2.1. Model component animation	16
2.2.2. Animation customization	17
2.2.3. Interactivity controls	18
2.3. Summary	19
3. Tools survey	22
3.1. Tool sample	22
3.2. Arena	22
3.3. Simul8	24
3.4. IBM WebSphere business modeler advanced 7.0	25
3.5. Progress savvion process modeler 8.0	26
3.6. Tibco business studio community edition	28
3.7. ProM UITopia	29

3.8.	Results summary	30
3.9.	Related work.....	32
3.10.	Summary	32
4.	OSS tool selection	34
4.1.	Introduction	34
4.2.	OSS tools evaluation framework.....	34
4.3.	Tools search and selection.....	36
4.3.1.	Search criteria.....	36
4.3.2.	Search results.....	37
4.4.	Description of the selected OSS projects	38
4.4.1.	Activiti.....	38
4.4.2.	Bonita open solution: Open source BPM	38
4.4.3.	BPMN2 Modeler	38
4.4.4.	jBPM	39
4.4.5.	Modelio – Modeling environment (UML)	39
4.4.6.	Yaoqiang BPMN editor.....	39
4.5.	Collected data from the OSS projects	40
4.6.	BPMN 2 elements coverage	41
4.7.	Fit for purpose evaluation criteria	42
4.7.1.	Source code access	42
4.7.2.	API flexibility.....	42
4.7.3.	Source code understandability.....	43
4.7.4.	Source code complexity	44
4.8.	Applying fit for purpose evaluation criteria	45
4.8.1.	Source code access results.....	45
4.8.2.	API flexibility.....	45
4.8.3.	Source code understandability.....	46
4.8.4.	Source code complexity	47

4.9.	Results summary	54
4.10.	Summary	55
5.	BPMN 2.0 elements animation.....	58
5.1.	Introduction	58
5.2.	BPMN 2.0 elements	58
5.3.	BPMN 2.0 proposed new elements	59
5.3.1.	Resources.....	59
5.3.2.	Queues.....	60
5.3.3.	Process instances	62
5.4.	Animation for BPMN 2.0 and proposed elements	63
5.4.1.	Fill color and border line color animation.....	63
5.4.2.	Border line thickness animation	64
5.4.3.	Events animation	64
5.4.4.	Activities animation	65
5.4.5.	Gateways animation	65
5.4.6.	Sequence flows animation.....	66
5.4.7.	Resources animation.....	66
5.4.8.	Queues animation	67
5.4.9.	Process instances animation	67
5.5.	Animation views.....	67
5.5.1.	Dynamic behavior view.....	67
5.5.2.	Path load analysis view	68
5.5.3.	Performance analysis view	68
5.6.	BPMN 2.0 elements graphical rules.....	69
5.7.	Summary	70
6.	Bpmn2Animator.....	72
6.1.	Introduction	72
6.2.	Bpmn2Animator use cases	72

6.3.	Plug-in development.....	76
6.3.1.	Plug-in dependencies.....	77
6.3.2.	Plug-in Extensions.....	79
6.3.3.	Bpmn2 Modeler runtime extension.....	81
6.4.	Bpmn2Animator package analysis.....	84
6.4.1.	Bpmn2Animator animation engines.....	86
6.4.2.	Bpmn2Animator model.....	95
6.4.3.	Bpmn2Animator user controls.....	96
6.4.4.	Bpmn2Animator event generators.....	101
6.5.	Summary.....	103
7.	Validation.....	106
7.1.	Introduction.....	106
7.2.	Business process – Request fulfillment.....	106
7.2.1.	ISCTE-IUL request fulfillment process.....	107
7.2.2.	Process roles.....	108
7.2.3.	Process activities.....	109
7.3.	Data preparation.....	110
7.3.1.	Data collection.....	110
7.3.2.	Data preparation.....	113
7.4.	Process execution data replay.....	115
7.4.1.	Service request fulfilment process replay.....	115
7.4.2.	11th of September, 2013.....	116
7.4.3.	19th of September, 2013.....	120
7.4.4.	26th of September, 2013.....	124
7.5.	Summary.....	128
8.	Conclusions and future work.....	130
8.1.	Conclusions.....	130
8.2.	Future work.....	132

Bibliography	133
Annexes.....	137
A. OSS Projects web addresses.....	137
B. Bpmn2Animator development	138
B.1 Bpmn2Animator extension elements of Bpmn2 Modeler.....	138
C. Service request fulfillment process data preparation.....	140
C.1 CSV Analyzer internals.....	140
C.2 Sort helper list	144
C.3 IT service management system actions description	145
C.4 Actions to ignore list content.....	146
C.5 Service request process groups.....	146
D. Service request process model parameterization.....	147
E. Service request fulfillment process replay results	150
E.1 11th of September of 2013	151
E.2 19th of September of 2013	153
E.3 26th of September of 2013	155

[This page is intentionally blank]

List of Figures

Figure 1 – Research activities and contributions.....	8
Figure 2 - Arena animation example.....	23
Figure 3 - Simul8 animation example.....	24
Figure 4 - Websphere animation example.....	26
Figure 5 - Savvion animation example.....	27
Figure 6 - Tibco animation example.....	28
Figure 7 - ProM UITopia animation example (source: (Aalst, Leoni and Hofstede 2012)).....	29
Figure 8 - Tool scores comparison histogram.....	31
Figure 9 - BPMN2 Modeler size analysis treemap view.....	47
Figure 10 - Modelio size analysis treemap view.....	48
Figure 11 - Yaoqiang size analysis treemap view.....	48
Figure 12 - BPMN2 Modeler complexity analysis view.....	49
Figure 13 - Modelio complexity analysis view.....	50
Figure 14 - Yaoqiang complexity analysis view.....	50
Figure 15 - BPMN2 Modeler CA analysis with dependency view.....	51
Figure 16 - Modelio CA analysis with dependency view.....	52
Figure 17 - Yaoqiang CA analysis with dependency view.....	52
Figure 18 - BPMN2 CE analysis with dependency view.....	53
Figure 19 - Modelio CE analysis with dependency view.....	53
Figure 20 - Yaoqiang CE analysis with dependency view.....	54
Figure 21 - BPMN metamodel extract related with resource assignment (source: (OMG 2011))......	59
Figure 22 - Graphical notation proposed for human resource roles.....	60
Figure 23 - Graphical notation proposed for automated resource roles.....	60
Figure 24 - BPMN metamodel extract related to activities (source: (OMG 2011))......	61
Figure 25 - Propose extension to the BPMN metamodel.....	61
Figure 26 - Proposed graphical notation for a Queue.....	62
Figure 27 - Instance metaclass.....	62
Figure 28 - Icon for human process instances.....	62
Figure 29 - Icon for document process instances.....	63
Figure 30 - Icon for default process instances.....	63
Figure 31 - Border line thickness and border line color change example.....	63
Figure 32 – Fill color change example.....	63
Figure 33 – Color spectrum example.....	64
Figure 34 – Start Event border line color and fill color animation example.....	65

Figure 35 - End Event border line color and fill color animation example.....	65
Figure 36 - Intermediate Event border line color, and fill color animation.....	65
Figure 37 - Examples of activity animation (<i>in the left: only border line thickness and border line color. In the middle: only fill color. On the right: combination of the three animation types</i>).....	65
Figure 38 - Gateway border line thickness, border line color and fill color animation.....	66
Figure 39 - Sequence flow line thickness and line color animation example	66
Figure 40 - Resource color animation type figure.....	66
Figure 41 - Resource states icons (<i>From left to right: “Idle”, “Busy” and “Fail”</i>).	66
Figure 42 - Example of resource presentation with its status.....	67
Figure 43 - Queue color animation example	67
Figure 44 - Bpmn2Animator use case diagram.....	72
Figure 45- Animation execution controls.....	73
Figure 46 - Start animation dialog.....	73
Figure 47 - Event Manager dialog.....	73
Figure 48 - Bpmn2Animator preferences window.....	74
Figure 49 - Bpmn2Animator animation display control view.....	75
Figure 50 - Task animation attributes property tab	75
Figure 51 - Bpmn2 Modeler dependencies	76
Figure 52 - Bpmn2Animator Eclipse contributions demonstration.....	77
Figure 53 - Bpmn2Animator preferences page	77
Figure 54 - Bpmn2Animator plug-ins dependencies.....	79
Figure 55 - Extension points used by Bpmn2Animator	80
Figure 56 - Bpmn2 Modeler preferences window with Target Runtime	81
Figure 57 - Extension elements from Bpmn2 Modeler used by Bpmn2Animator.....	82
Figure 58 - <i>HumanPerformer CustomTask</i> extension details	83
Figure 59 - Bpmn2 Modeler palette with the “Animator” category.....	83
Figure 60 - <i>HumanPerformer</i> usage example.	84
Figure 61 - Bpmn2Animator package diagram (all packages).....	86
Figure 62 - Package dependencies diagram centered on the engines package.....	86
Figure 63 - Engines package class diagram.	88
Figure 64 - <i>AnimationEngine</i> interaction diagram	88
Figure 65 - Sequence diagram of the <i>AnimationEngine updateElement</i> method.....	89
Figure 66 - <i>diagramAnimation</i> package class diagram.	89
Figure 67 - Sequence diagram for <i>updatePictogram</i>	90
Figure 68 - <i>DiagramAnimationFeatureExecuter</i> interaction diagram	90

Figure 69 - <i>DiagramAnimationEngine</i> interaction diagram.....	91
Figure 70 - Process instance animation class diagram	92
Figure 71 - Process instance animation interaction diagram.....	93
Figure 72 - Instance animation sequence diagram	93
Figure 73 - <i>ModelMetrics</i> package class diagram.....	94
Figure 74 - <i>ModelAnimationEngine</i> interaction diagram.....	95
Figure 75 - EMF model editor.....	96
Figure 76 - <i>Bpmn2AnimatorModel</i> package diagram	96
Figure 77 - User controls centered package view	97
Figure 78 - <i>Preferences</i> package class diagram.	99
Figure 79 - <i>Handlers</i> package class diagram.	99
Figure 80 - <i>Views</i> package class diagram.....	100
Figure 81 - <i>EventGenerators</i> package dependencies diagram.	101
Figure 82 - <i>EventGenerators</i> package class diagram.	102
Figure 83 - <i>EventGenerator</i> interaction diagram	103
Figure 84 - BPMN model of the request fulfillment process	108
Figure 85 - IT Management System data export web page (Operation -> Actions -> All Actions) ...	111
Figure 86 - IT Service Management System - Filter web interface	112
Figure 87 - Sorting criteria to prepare the file for CSV Analyzer plug-in	113
Figure 88 - CSV Analyzer options window	114
Figure 89 - Service requests fulfilment process activity graphic	115
Figure 90 - Process status 11th September (<i>presented metrics: Activities - num. of proc. instances; Performers - num. of proc. instances; Queues - num. instances that waited in the queue</i>)	118
Figure 91 - Process status 11th September (<i>presented metrics: Activities - num. of SLA violations; Performers - total of proc. time; Queues - wait time</i>).....	119
Figure 92 - Process status 11th September (<i>presented metrics: Activities - num. of SLA violations; Performers - total of proc. time; Queues - Avg. wait time per instance</i>)	120
Figure 93 - Process status 19th September (<i>presented metrics: Activities - num. of proc. instances; Performers - num. of proc. instances; Queues - num. of instances that waited in the queue</i>)	122
Figure 94 - Process status 19th September (<i>presented metrics: Activities - num. of SLA violations; Performers - total of proc. time; Queues - total of wait time</i>).....	123
Figure 95 - Process status 19th September (<i>presented metrics: Activities - num. of SLA violations; Performers - total of proc. time; Queues - Avg. wait time per instance</i>)	124
Figure 96 - Process status 26th September (<i>presented metrics: Activities - num. of proc. instances; Performers - num. of proc. instances; Queues - num. of instances that waited in the queue</i>)	126

Figure 97 - Process status 26th September (<i>presented metrics: Activities - num. of SLA violations; Performers - total of proc. time; Queues - total of wait time</i>).....	127
Figure 98 - Process status 26th September (<i>presented metrics: Activities – num. of SLA violations; Performers – total of proc. time; Queues – Avg. wait time per instance</i>).....	128
Figure 99 - CSV Analyzer main method UML sequence diagram	141
Figure 100 - applyRulesToEvents UML sequence diagram	142
Figure 101 - Service request process model.....	148

List of tables

Table 1 - Design science research guidelines. Source: (Hevner, March, Park and Ram 2004).	5
Table 2 – Research contributions by artifact type	6
Table 3 - Research contributions grouped by contribution type	7
Table 4 - Queue, Resource and Activity animation	16
Table 5 - Process instance animation	17
Table 6 - Sequence flow animation.....	17
Table 7 - Attributes and global variables animation	17
Table 8 - Queue and Activity customization.....	17
Table 9 - Resource customization	18
Table 10 - Process instance customization.....	18
Table 11 - Sequence flow customization.....	18
Table 12 - Attributes and global variables customization	18
Table 13 - Animation speed controls.....	19
Table 14 - Animation interaction controls.....	19
Table 15 - Animation visualization controls	19
Table 16 - Tool sample.....	22
Table 17 - ARENA evaluation scores	24
Table 18 - Simul8 evaluation scores	25
Table 19 - Websphere evaluation score.....	26
Table 20 - Savvion evaluation score	27
Table 21 - Tibco evaluation score	28
Table 22 - ProM UITopia evaluation score.....	30
Table 23 - Summary of scores for component animation	30
Table 24 - Summary of scores for animation customization.....	30
Table 25 - Summary of scores for animation interactivity.....	31
Table 26 - Final rank of surveyed tools regarding animation features.....	31
Table 27 - OSS projects success evaluation criteria.....	34
Table 28 - Number of results by open source projects repository	37
Table 29 - OSS projects found in the search process	38
Table 30 - Selected OSS projects data	40
Table 31 - BPMN 2 coverage results	41
Table 32- Source code access easiness.....	42
Table 33 - API flexibility classification criteria	43
Table 34 - Javadoc coverage evaluation criterion	44

Table 35 - Source code access results	45
Table 36 - API requirements list	45
Table 37 - API flexibility results	46
Table 38 - Number of source code languages results.....	46
Table 39 - Javadoc coverage results.....	47
Table 40 - OSS tools analysis summary.....	55
Table 41 - BPMN 2.0 Basic elements notation (source: (OMG 2011)).....	58
Table 42 - Resource graphical notation used by BPMN task type.....	60
Table 43 - Text, color, size and lines rules for BPMN.....	69
Table 44 - Bpmn2 Modeler plug-ins	78
Table 45 - Graphiti plug-ins	78
Table 46 - EMF plug-ins	78
Table 47 - Eclipse IDE plug-ins	78
Table 48 - Bpmn2Animator used Extension points and their description.....	80
Table 49 - org.eclipse.bpmn2.modeler.runtime extension elements description.	82
Table 50 - <i>Bpmn2Animator</i> package descriptions.....	84
Table 51 - <i>Preferences</i> package classes' description	99
Table 52 - <i>Handlers</i> package classes description.....	100
Table 53 - Classes descriptions	100
Table 54 - Request Fulfillment process activities	107
Table 55 – Request fulfillment process roles	108
Table 56 - Columns selected in the view.....	112
Table 57 - Columns added in the first stage of the data preparation	113
Table 58 - Bpmn2Animator evaluation results	131
Table 59 - Bpmn2Animator score	131
Table 60 - OSS projects web addresses.....	137
Table 61 – Bpmn2Animator extension elements	138
Table 62 - Applying data preparation rules methods description.....	142
Table 63 - SortHelper content	145
Table 64 - IT service management system actions descriptions	145
Table 65 - Actions to ignore.....	146
Table 66 - Groups defined in the IT service management system	147
Table 67 – Group levels mapping with process model lanes	148
Table 68 - Gateway 1 (“to 1st line?”) conditions.....	148
Table 69 - Gateway 2 (“is to proceed?”) conditions	149

Table 70 - Gateway 3 (“to 1st line?”) conditions	150
Table 71 - Gateway 4 (“to proceed?”) conditions	150
Table 72 - Gateway 5 (“is validated?”) conditions	150
Table 73 - Queues results	151
Table 74 - Performers results	151
Table 75 - Gateways results Table 76 - Boundary events results.....	151
Table 77 - Gateways results	152
Table 78 - Boundary events results	152
Table 79 - Sequence flows results	152
Table 80 - Activities results.....	153
Table 81 - Queues result.....	153
Table 82 - Performers results	153
Table 83 - Gateways results	154
Table 84 - Boundary events results	154
Table 85 - Sequence flows results	154
Table 86 - Activities results.....	155
Table 87 - Queues results	155
Table 88 - Performers results	155
Table 89 - Gateways results	156
Table 90 - Boundary events results	156
Table 91 - Sequence Flows results	156

[This page is intentionally blank]

Chapter 1

Introduction

Contents

1. Introduction	2
1.1. Motivation	2
1.2. Research problem	2
1.2.1. Problem statement	2
1.2.2. Research questions	2
1.3. Expected contributions and objectives	3
1.4. Research methodology	4
1.5. Business process models animation	9
1.6. BPMN – Business Process Modeling Notation	10
1.7. Document structure	11

1. Introduction

1.1. Motivation

Animation can be defined as a dynamic presentation of objects that change position, form or color on top of a static background (Pegden et al. 1995). In the case of process model animation, the static background is the process model itself. Basically, an animated layer is added on top of the static process model in order to show information about its execution. Using animation one can understand the process dynamic behavior while watching a movie presenting the process execution events. The movie may present, for example, the resources usage, the process instances travelling through the process paths or simply the activities usage. Process models animation can be used for designing/redesigning business processes or for monitoring the processes execution performance.

The usage of animation to illustrate processes performance appeared first in general purpose simulation tools like arena (Automation 2014). Animation has gained importance since then, and nowadays it can be found in many general purpose simulation tools, as well as in Business Process Management dedicated tools. The problem with process models animation is that it varies much from tool to tool. This can be understandable in the case of general purpose simulation tools since many of them have its own modeling language and therefore its own way of animating the processes. However this becomes less understandable and desirable in the case of Business Process Management dedicated tools, since most of them use the Business Process Model and Notation (BPMN) in order to model the business processes. We believe that a scientific effort should be made in order to present a BPMN extension that can be used to animate business processes and give one step forward towards a BPMN animation standard.

1.2. Research problem

1.2.1. Problem statement

How to animate BPMN models to present business processes execution information?

The problem addressed in this work has to do with the BPMN notation and in which way the notation can be extended in order to support the animation of business process models.

1.2.2. Research questions

Following the problem formulation there were some research question that were used to guide our work:

How is the animation supported by the commercial tools?

Which process elements are animated and how are they animated?

Which Open Source Software tool should we use as basis for the development of our prototype?

How to manipulate the BPMN elements without violating the notation rules?

1.3. Expected contributions and objectives

In the end of our work we expect to have the following contributions:

- Propose and apply a taxonomy for tool animation support assessment.

In order to assess the state of the art of business process models animation, we intend to gather a sample of commercial tools that present state of the art animation features. In order to ensure comparability of the commercial tools we searched for a research work that presented a taxonomy that we could use. However, because we were not able to find one, we intend to develop one and apply it to the sample of tools in order to draw conclusions about the state of the art of business process models animation. The taxonomy should be able to evaluate the animation support of each element as well as the animation customization and the tool animation interaction capabilities.

- Application of the developed taxonomy in order to understand the state of the art.

After developing our taxonomy we intend to apply it to a sample of animation supporting tools in order to understand the state of the art of process model animation.

- Propose a BPMN extension that endows the notation with process model animation features.

We intend to develop a BPMN extension that shows how the notation elements may be animated in order to present process execution details.

- Develop an Open Source Software prototype that implements the BPMN extension.

We also intend to develop a software prototype that is able to read process execution events from a log and animate the process model at same time using our BPMN extension.

- Propose a framework for Open Source Software tools selection

Our developed prototype will be developed upon other Open Source Software (OSS) tool that supports business processes modeling in BPMN, and also fits our development requirements. Since we were not able to find one framework that could facilitate the choice of such tool, we intend to propose one and apply it in a sample of OSS tools.

1.4. Research methodology

Our research follows the design science research methodology. According to Hevner et al. (2004), design science is a problem solving paradigm. It seeks to extend the boundaries of human and organizational capabilities through the creation of new and innovative artifacts. In this paradigm the knowledge and understanding of a problem domain, as well as its solution are achieved in the building and application of the designed artifact. Hevner et al. (2004) state that IT artifacts are broadly defined as constructs (vocabulary and symbols), models (abstractions and representations), methods (algorithms and practices) and instantiations (implemented and prototype systems).

Constructs: provide the language in which problems and solutions are defined and communicated (Shön 1983).

Models: use constructs to represent a real world situation – the design problem and its solution space (Simon 1996).

Methods: define processes and provide guidance on how to solve problems, that is, how to search the solution space (Hevner, March, Park and Ram 2004).

Instantiations: show that constructs, models, or methods can be implemented in a working system. They enable researchers to learn about the real world, how the artifact affects it, and how users appropriate it (Hevner, March, Park and Ram 2004).

Guidelines for design science in information systems research

In order to understand, execute and evaluate our research we followed the guidelines for Information systems research, presented in (Hevner, March, Park and Ram 2004). Here we describe the seven guidelines, presented in Table 1, and show how each guideline applies to our research.

Table 1 - Design science research guidelines. Source: (Hevner, March, Park and Ram 2004).

Guideline	Description
Guideline 1: Design as an artifact	Design-science research must produce a viable artifact in the form of a construct, a model, a method, or an instantiation.
Guideline 2: Problem relevance	The objective of design-science research is to develop technology-based solutions to important and relevant business problems.
Guideline 3: Design evaluation	The utility, quality, and efficacy of a design artifact must be rigorously demonstrated via well-executed evaluation methods.
Guideline 4: Research contributions	Effective design-science research must provide clear and verifiable contributions in the areas of the design artifact, design foundations, and/or design methodologies.
Guideline 5: Research rigor	Design-science research relies upon the application of rigorous methods in both the construction and evaluation of the design artifact.
Guideline 6: Design as a search process	The search for an effective artifact requires utilizing available means to reach desired ends while satisfying laws in the problem environment.
Guideline 7: Communication of Research	Design-science research must be presented effectively both to technology-oriented as well as management-oriented audiences.

Guideline 1

Hevner et al. (2004) state, regarding this guideline, that “*the result of design-science research in IS is, by definition, a purposeful IT artifact created to address an important organizational problem*”. The authors also indicate, in the same work, that the artifacts constructed in design science research are not required to be a full-grown information system and that the artifact instantiation demonstrates feasibility of the designed product.

Our research work main contribution is an Open Source Software prototype that implements our proposed BPMN extension, which endows the notation with animation capabilities. According to the definitions of artifact presented before, our prototype fits in the instantiation artifact, meaning that the guideline is respected in our research. Besides the instantiation artifact, our research contributes with two methods. First, the taxonomy on tool support for process animation, which can be used to assess the animation features of a modeling tool. The second method is an Open Source Software tools

selection framework that can be used to facilitate the process of choosing an OSS to extend with new functionalities in order to make the prototype development faster, since there is no need to develop everything from zero. All research contributions are presented in Table 2.

Table 2 – Research contributions by artifact type

Artifact type	Contribution
Constructs	BPMN Extension to endow the notation with animation features.
Models	Business process model of the request fulfillment process as is implemented in ISCTE-IUL.
Methods	Taxonomy on tool support for process animation. Open Source Software tools selection framework.
Instantiations	Bpmn2Animator prototype.

Guideline 2

As mentioned in (Hevner, March, Park and Ram 2004), “*the objective of research in information systems is to acquire knowledge and understanding that enable the development and implementation of technology-based solutions to heretofore unsolved and important business problems*”. In order to accomplish this objective, design science proposes the construction of innovative artifacts with the goal of changing the phenomena that occur.

Our problem has to do with the extension of BPMN notation, so it can support business process models animation. Nowadays, the tools that are focused in BPMN and support animation do not show animation capabilities as sophisticated as other general purpose simulation tools with proprietary process modeling language. On the other hand, each BPMN focused tool shows different animation capabilities, as well as different ways of animating the notation elements. Our goal is to propose an extension to the notation that can be implemented in a prototype. The extension should enable a BPMN focused tool to present animation capabilities that are more similar to the ones presented by general purpose simulation tools.

Guideline 3

Evaluation includes the integration of the artifact within the technical infrastructure of the business environment. One of the evaluation methods presented in (Hevner, March, Park and Ram 2004) is the case study method that we used to evaluate our research work. Using this evaluation method we intend to verify the utility, quality and efficacy of our developed prototype.

The case study evaluation method was used in our research to validate the prototype, and thus, our proposed BPMN extension. In order to validate the proposed taxonomy on tool support for process animation we used it to evaluate a sample of tools. Also, the taxonomy was submitted and presented in the International Conference on Exploring Services Sciences 1.3 (IESS) international conference. The OSS tools selection framework was also used to evaluate a sample of tools to validate its usefulness. All of the presented research contributions were presented in an annual workshop that is promoted by the Quantitative Approaches on Software Engineering And Reengineering (QUASAR) research group, where they were evaluated and criticized by a panel of experienced researchers.

Guideline 4

“Effective design-science research must provide clear contributions in the areas of the design artifact, design construction knowledge (i.e., foundations), and/or design evaluation knowledge (i.e., methodologies)”. Three types of research contributions are identified: The design artifact, foundations and methodologies. The first one is the design artifact itself. Foundations are evaluated constructs, models, methods, or instantiations that extend the existing foundations in the design science, for example a modeling formalism. Methodologies represent the creative development and use of evaluation methods (Hevner, March, Park and Ram 2004).

Our expected contributions are presented in Table 3 grouped by research contribution type. These contributions are described in more depth in section 1.3.

Table 3 - Research contributions grouped by contribution type

Contribution type	Contribution description
Foundations	- BPMN extension that endows the notation with process model animation features.
Methodologies	- Taxonomy for tool animation support assessment. - Framework for Open Source Software tools selection.

Design artifact	- Open Source Software prototype that implements the BPMN extension.
-----------------	--

Guideline 5

“Design-science research requires the application of rigorous methods in both the construction and evaluation of the designed artifact” (Hevner, March, Park and Ram 2004).

To ensure the rigor in our research work, we submitted our taxonomy on tool support for process animation to be evaluated by the reviewers of the IESS1.3 conference in order to get an experts validation and get inputs for our work. An evaluation and validation by peers was also done on three annual workshops organized by the Quasar research group, where we presented our research contributions. The inputs gathered on these workshops contributed positively to our research rigor.

Guideline 6

“Design is essentially a search process to discover an effective solution to a problem” (Hevner, March, Park and Ram 2004).

To develop our BPMN extension proposal we started by studying the notation elements, analyzing which elements should, and how they could be animated. We used all knowledge acquired from our tool survey, and respected all presentation rules of the notation elements in the conception of our extension. It was then implemented in a software prototype and finally several examples of an animated process model were presented in the case study. A model representation of our research work activities and their main contributions is presented in Figure 1.

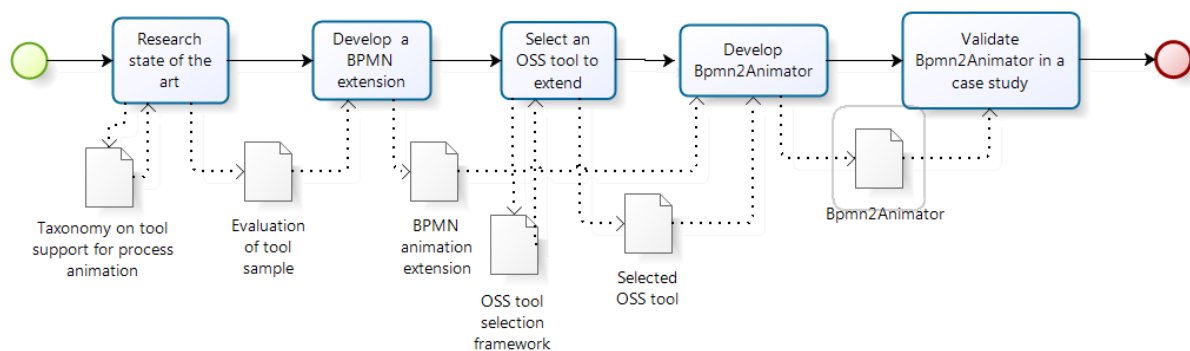


Figure 1 – Research activities and contributions

Guideline 7

“Design science research must be presented both to technology-oriented as well as management-oriented audiences” (Hevner, March, Park and Ram 2004).

Our research work tries to address the problem of process models animation for both identified audiences. We tackle the problem from a management point of view, afterwards, when introducing the audience to business process models animation. Next, tools that support process models animation are surveyed and their animation features evaluated with an effort to keep understandable for both audiences. The next chapters are more focused on the technical part of our research, making those chapters more directed to a technical-oriented audience. Finally, we present our research evaluation through a case study that is aimed for both audiences. Our research work was presented in three annual workshops promoted by the QUASAR research group, and our proposed taxonomy on tool support for process animation and its application on a sample of tools were presented in the IESS 1.3 international conference. In both QUASAR workshops and IESS 1.3 conference the audience was composed by technology-oriented researchers as well as management-oriented researchers.

1.5. Business process models animation

Business process models can be animated to show process execution events information. The process execution events can have different sources, for instance, they can be provided by a simulation tool or by a business process execution engine. The events from an execution engine can be used to animate the process model in real time, so the viewer can monitor the process. On the other hand, its events can be exported from the engine and be used to replay the business process execution through process model animation. The model animation presents a movie of the model execution in which it is possible to watch the model components interacting, like for example the model instances waiting in the activities queues or the resources working and processing the instances. This animation can help to better understand the dynamic behavior of the model. In (Pegden, Shannon and Sadowski 1995) animation is defined as a dynamic presentation of objects that change position, shape or color on top of a static background. As mentioned before, animation can be useful for models verification and validation (Sargent 1998), however other usages for animation are referred in the literature. In (Rohrer 2000) animation is referred as being useful for the communication of large amounts of information and in (Agarwal 2011) is mentioned that animation can deliver complex messages that would be difficult to deliver through simulation. The animation capability to facilitate communication is also pointed in (Agarwal 2011; de Vreede and Verbraeck 1996; Janssen et al. 2009; Rohrer 2000). In (de Vreede and Verbraeck 1996) the authors conclude that “*animation together with statistical sound data can make the dynamic nature of a problem situation explicit in such a way that the problem owners or decision makers will draw valid conclusions or rightfully demand further research*“. Other benefit of animation is the fact that it facilitates the sharing of complex ideas and helps the creative process of developing new solutions for the problems (de Vreede and Verbraeck 1996; Rohrer 2000). In (Janssen,

Joha and Zuurmond 2009) is presented a case study where simulation was used together with animation to understand how the adoption of shared services by the Dutch public agencies could improve their efficiency and service levels. The goal for the use of animation was to make the communication easier between the stakeholders, so they could better understand the process and evaluate it qualitatively. In the conclusions section, the authors claim that *“for some workshop participants, the main benefits were related to the facilitation of ideas, creating understanding, and discussing the results by viewing the animation, instead of using simulations for a detailed quantitative analysis”*.

1.6. BPMN – Business Process Modeling Notation

The BPMN notation was created in 2001 by BPMI.org (Business Process Management Initiative), which had the goal of creating a XML language for business process execution. In the creation process BPMI.org ended discovering the need for a graphical representation for the business processes. Several commercial companies with interest in business process modeling participated in the notation development process. They shared the same goal of creating an unique standard notation that could be adopted by several tools (White and Miers 2008). Meanwhile, BPMI.org became part of OMG (Object Management Group), which was responsible for the development of the UML (Unified Modeling Language) (Allweyer 2010).

The last version made available was the 2.0 in 2010. The objective for the notation is still the same, to create a notation that is easily understood by every business user, as well by the technical developers responsible for the development of the process execution tools. The second objective is to assure that languages developed for business process execution, like for example BPEL (Business Process Execution Language) can be visualized through a notation that is business oriented (Silver 2009). The main benefits of using the BPMN notation are the following.

- BPMN is a standard with a public specification including a metamodel that can be extended in order to enhance the notation with more specific capabilities, like for instance animation ones (Silver 2009);
- BPMN does not have ambiguity in its notation components. This means that each graphical element has a specific meaning, as well as specific modeling rules, which makes the communication of process details easier, since we only have to know the notation and we will easily understand the modeled processes without further explanation;
- BPMN can be translated to a business process execution language like BPEL.

In (Correia 2014) an assessment of business process modelling languages based on a proposed taxonomy is presented and it is concluded that BPMN is nowadays the most well equipped process modelling language.

1.7. Document structure

This document contains a brief introduction on process models animation and BPMN notation. Next follows a chapter where the taxonomy for animation tool support is presented together with its application to study a sample of commercial tools. Next, a framework for Open Source Software tools is presented and applied to a sample of Open Source Software tools. After this chapter, an extension of BPMN notation is presented followed by a chapter where our prototype is presented. A case study is then presented followed by conclusions, future work and bibliographical references.

[This page is intentionally blank]

Chapter 2

A taxonomy on tool support for process animation

Contents

2. A taxonomy on tool support for process animation	14
2.1. Business Process Simulation	14
2.2. The taxonomy categories.....	16
2.3. Summary	19

In this chapter the benefits and limitations of Business Process Simulation are introduced. Also, a taxonomy to assess the animation support of each element as well as the animation customization and the tool animation interaction capabilities is proposed.

2. A taxonomy on tool support for process animation

2.1. Business Process Simulation

Business Process Simulation is a technique that allows the representation of processes, people and technology in a dynamic computer model (Harrington 1991). BPS (Business Process Simulation) is also referred as “discrete event simulation” and has its roots in the manufacturing systems analysis. Later this technique was used to assist in the management of change of manufacturing and services settings (Greasley 2000). BPS is addressed in Business Process Management, which is identified as being an integrated system to manage the business performance through the management of end-to-end business processes (Hammer 2010). A business process is a logically related group of activities that use organization resources to produce results which support the organization objectives (Harrington 1991). There are reports of BPS usage in business processes reengineering projects like, for example, (Greasley 2000) and (Dennis et al. 2000). In these two particular cases, simulation is used in the design phase of the BPM lifecycle where the to-be process model is conceived. Tumay (1995) stated that business processes are too complex and too dynamic to be analyzed and understood through the use of techniques like flowcharting or spreadsheets. For that reason, organizations should use simulation in business process reengineering, because it provides analysis with precision, providing also the visualization of alternatives. With the information of probabilities at decision points and probability distributions defining activity durations Business Process Simulation allows predicting process performance along a number of measures such as lead-time¹, resource utilization and cost (Greasley 2000).

2.1.1. BPS benefits

Business process simulation presents several benefits which help justifying the interest and applicability in business process management. Greasley (2000) presents the following three benefits:

Risk Reduction – *“The use of scenarios and sensitivity analysis helps reducing the unknown risk.”*

Proves Concept – *“The model of the proposed process can help sell and ‘unfreeze’ current ways of doing. The use of animation can help understanding and the use of data collection of performance measurement can help management by ‘facts’.”*

Identifies change strategies – *“The simulation can assess process change from the perspectives of a change in demand on the process, a change in resource availability on the process and a change in the design of the process.”*

¹ Lead-time is the time from the start of the first operation to the completion of the last one (Kelton et al. 2002).

More benefits are presented in (Indihar-Stemberger et al. 2003), where the authors state that the main benefit of simulation modeling is the fact that it combines analysis and assessment in qualitative or quantitative terms. The following benefits are also presented in the same work:

What-if scenarios – Through simulation it is possible to experiment several what-if scenarios, where each one can go from a different combination of resources to a different process design with a different statistical distribution used for the decision points, which result in different performance, times and cost outputs. It is through the comparison of several scenarios that is possible to find the scenario with the best output performance. In (Hlupic and Robinson 1998) it is mentioned that alternative process designs can significantly improve the chance of a successful business process reengineering project, which can then result in capital cost reduction, better efficiency of processes, better service provided to customers and/or lead time reduction.

Dynamic behavior modeling and presentation – Nowadays simulation tools are able to model the process dynamic component and present it visually. This type of presentation is facilitated by graphical interfaces that are able to animate and display graphically the simulation results. In (Hlupic and Robinson 1998) the authors state that visual interactive simulation models, together with a variety of graphical output reports, can demonstrate the benefits of redesigned processes.

Experimenting before implementation – Experimenting with a simulation model, rather than implementing directly in real process, reduces the risk of making wrong decisions (Hlupic and Robinson 1998).

2.1.2. BPS limitations

Although Business Process Simulation has many benefits it also has some drawbacks to be explored. One of the drawbacks is the fact that the human resources are too simplified, since it is not possible to model correctly, and in a real way, the human behavior (Aalst 2010). This problem derives from the fact that people (1) are usually involved in several processes; (2) do not work in a constant rhythm and (3) tend to work in part-time, dedicated to one function at a time, and do not follow a predicted order of work (Aalst 2010). Besides these points, it is hard to model the work priorities and the changes that the process suffers under determined contexts (Aalst 2010).

Other limitation of common simulation tools is that simulation parameters like activity execution times, waiting times, or resource availability are configured manually depending on data gathered before (Indihar-Stemberger, Popovic and Bosilj-Vuksic 2003). However, there are scientific works that go in the direction of searching a solution for this problem through the usage of data gathered from information systems which are the base for the existing processes in an organization, like Enterprise Resource Planning (ERP) systems or others transactional information systems. The idea is

to use event logs as the source of information for the simulation parameterization and then apply variability through the use of statistical distributions fitted for the events (Wynn et al. 2008), (Aalst 2010).

On the other hand, there is a problem that is not explained by the lack of capacity of the simulation tools or by the simulation itself, but ends limiting the simulation results. The problem can be found in the simulation models that are created and used. Sometimes, the quality of some models is not good which can be explained by (1) incorrect process modeling or (2) by the shortage of data gathered for the creation and parameterization of the model (Aalst 2010; Terrence and Kapila 2001). There are verification and validation techniques that can be applied in order to minimize the problems that appear, because the model does not represent correctly the reality. As mentioned in (Sargent 1998). Verification is the continuous process of assuring that the model works as it should, whereas the validation is the process of assuring that the model represents the reality (Chung 2003). In both articles (Chung 2003) and (Sargent 1998) simulation models animation appears as being a technique that can be used for both model verification and validation.

2.2. The taxonomy categories

This taxonomy combines several criteria organized in categories: *model component animation*, *animation customization* and *interaction controls*, which will be described hereafter. Each criterion within a category is expressed on an ordinal scale, where the lowest score (1) is assigned if the evaluated feature is unavailable and the highest score (4) corresponds to the best known state of the art.

2.2.1. Model component animation

This group of criteria (*queue, resource and activity animation, process instance animation, sequence flow animation, attributes and global variables animation*) evaluates tool capabilities to animate each modeling component separately. This kind of animation is usually shown close to the component, to represent the modifications of a given attribute. The following tables describe the corresponding grading scales.

Table 4 - Queue, Resource and Activity animation

1	No support
2	Numeric values only: Numbers are placed near the corresponding activity.
3	Bar with shape and color changes: A bar is shown near the activity and its size and/or color changes over time.
4	Animated figure: An animated figure changes its appearance.

Table 5 - Process instance animation

1	No support
2	Indicative default figure animation: Represents a false type of animation, since it does not show the passage of a process instance through the sequence flow. Instead, the animation only appears if a sequence flow arrow is selected in order to highlight it from the rest of the model.
3	Default figure animation: The process instances are animated throughout process execution, but are all presented with an equal figure.
4	Fully animated figure: Process instances change their representation during process execution in order to express different roles, like a client or a document.

Table 6 - Sequence flow animation

1	No support
2	Color change only at process instance passage: The sequence flow changes its color to show the passage of a process instance.
3	Color change: The sequence flow can be animated with several colors, each with a specific meaning. This can be used to represent if a sequence flow path has more process instances trips than other paths of the model (e.g. with a brighter color).
4	Color and form change: The sequence flow arrows can become wider or thinner and change its color. The sequence flow arrows can become wider with the passage of process instances and its color can also become brighter. This can be useful to distinguish paths in the process that have more traffic than others.

Table 7 - Attributes and global variables animation

1	No support
2	Numeric values only: Attribute or global variables values are shown during process execution.
3	Temporal evolution graphics: Graphics displaying the evolution of attribute or global variables value over time may be shown.
4	Multi variable/attribute graphics: Graphics combining the values of various attributes or/and global variables may be shown.

2.2.2. Animation customization

The animation customization group of criteria (*queue and activity customization, resource customization, process instance customization, sequence flow customization, attributes and global variables customization*) relates to the capability of choosing how the model components are animated. The grading scales for each of these customizability criteria are described in the following tables.

Table 8 - Queue and Activity customization

1	No support
2	Customization of presented numeric values / bars: We may choose which component attribute is shown as numeric value and/or animated bar.
3	Customization of animated figure: We may customize an animated figure next to the component.
4	Cumulative customization: Both the customization of presented numeric values/bars (grade 2) and animated figures (grade 3) are supported.

Table 9 - Resource customization

1	No support
2	Customization of presented numeric values / bars and representative figure: We may choose the representative static figure for the resource and also which resource attribute is shown as numeric value and/or animated bar.
3	Customization of animated figure: We may choose the representative figure for each resource state (e.g.: busy, fail or idle).
4	Cumulative customization: We may customize the presented numeric values / bars and representative figure (grade 2), as well as the animated figure (grade 3).

Table 10 - Process instance customization

1	No support
2	Customization of representative figure: We may choose the representative figure for process instances.
3	Customization of figure by sequence flow: We may choose the representative figure of process instances for each sequence flow.
4	Customization of figure changing logic: We may specify a logic that defines which figure should be shown for each process instance. For instance, the figure may change depending on the value or range of an attribute.

Table 11 - Sequence flow customization

1	No support
2	Customization of color: We may choose the color that is presented when a process instance passes in the sequence flow.
3	Customization of color changes: We may choose how the sequence flow changes color. It is possible to choose the colors, the attribute and attribute value ranges corresponding to each sequence flow color.
4	Customization of color and shape changes: It is possible to customize the change in sequence flow thickness and color.

Table 12 - Attributes and global variables customization

1	No support
2	Customization of numeric values: We may choose attributes and global variables to be presented in a chosen position of the model as a numeric value.
3	Customization of graphics: We may customize graphics that represent attributes and global variables.
4	Cumulative customization: We may customize the numeric values (grade 2), as well as the graphics (grade 3).

2.2.3. Interactivity controls

The interactivity controls group of criteria (*animation speed control*, *animation interaction control*, *animation visualization control*) includes the type of controls that enable the user to interact with process execution.

Table 13 - Animation speed controls

1	No support
2	Play controls: We may play, pause, resume and stop process execution.
3	Speed controls: We may execute the process faster or slower and go directly to a chosen date/time.
4	Cumulative control: We may use both the play (grade 2) and speed (grade 3) controls.

Table 14 - Animation interaction controls

1	No support
2	Path choice control: We may flow through the model, following a process instance and deciding where the instance should go in the gateways.
3	Visual interactive animation control: We may control model components by manipulating their attributes, as well as global variables. This type of animation helps understanding model behavior and validate it (Rekapalli and Martinez 2007; Rohrer 2000).
4	Cumulative control: We may use the path choice control (grade 2), as well as the visual interactive animation control (grade 3).

Table 15 - Animation visualization controls

1	No support
2	Zoom and viewport: We may zoom and change the viewport.
3	Animation filter: We may choose which components to be animated.
4	Cumulative control: We may use the zoom and viewport features (grade 2), as well as the animation filter (grade 3).

2.3. Summary

In this chapter a taxonomy that can be used to assess the tool support for process animation was presented. This taxonomy was developed in order to be used to research the state of the art in process models animation. It was developed because we were not able to find a suitable taxonomy that could be used to draw conclusions about animation features support. The proposed taxonomy is divided in three categories; model component animation; animation customization; and interactivity controls. Each one of them with several evaluation criteria.

[This page is intentionally blank]

Chapter 3

Tools survey

Contents

3. Tools survey	22
3.1. Tool sample	22
3.2. Arena	22
3.3. Simul8	24
3.4. IBM WebSphere business modeler advanced 7.0	25
3.5. Progress savvion process modeler 8.0	26
3.6. Tibco business studio community edition	28
3.7. ProM UITopia	29
3.8. Results summary	30
3.9. Related work	32
3.10. Summary	32

In this chapter a sample of tools that support process animation is presented. Each tool is then presented individually. In the end we use the taxonomy to evaluate each tool and compare the results of each tool to draw conclusions.

3. Tools survey

3.1. Tool sample

Relevance, diversity and availability were the three basic criteria used in the tool sample selection. Relevance, in this context, stands for the availability of rich model animation features. We discarded tools that missed the description of those features in their online documentation. Regarding *diversity*, we tried to look at diverse origins, instead of limiting the survey to a limited target (e.g. just simulation tools) or modeling notation (e.g. just BPMN tools). Last, but not the least, *availability* had a considerable influence on our tool sampling process, since the only way to effectively evaluate a tool is by being able to obtain a fully functional version of it for a sufficient period of time. Several apparently interesting tools had only limited functionality trials or could not be obtained easily for evaluation in due time and therefore were discarded.

The chosen sample of six tools (see Table 16) meets the previous criteria. Since model animation features are used for planning, design, optimization and reengineering of real production, manufacturing, logistic or service provision systems that can be found in most medium to large size companies, it came as no surprise that commercial tools dominate this sample.

Table 16 - Tool sample

TOOL (VERSION) / PRODUCER	NOTATIONS	SCOPE
Arena (14) / Rockwell Automation	Proprietary	Simulation
SIMUL8 (2012) / SIMUL8 Corporation	Proprietary, BPMN	Simulation
WebSphere Business Modeler Advanced (7.0) / IBM	BPMN	Modeling
Savvion Process Modeler (8.0) / Progress	BPMN	Modeling
TIBCO Business Studio (Community Edition, 2012) / TIBCO Soft.	BPMN	Modeling
ProM UITopia (6) / Eindhoven Technical University	Petri nets, EPC, BPMN	Mining

Each tool will now be reviewed, in a separate subsection, regarding the aforementioned animation features. To facilitate the assessment exercise, an animation example is presented, which has a set of numbers placed on top of it, to identify model components animations. The evaluation scores are presented in the end of each subsection.

3.2. Arena

Arena is a general-purpose simulation tool that can be used in diverse systems, such as assembling lines and call centers. It has rich animation capabilities and is well-established in the marketplace, with several books dedicated to its usage (Altiok and Melamed 2007; Chung 2003; Kelton et al. 2002). It uses a straightforward proprietary notation where modeling constructs map easily to those described in chapter 2.2.1.

In Figure 2 is possible to identify the queues (1) filled with several process instances on top of an activity, as well as the process instances (3) that travel a sequence flow. It is also possible to see a global variable (6) exposed as a numeric value, a graphic representation (7) of a global variable and the combination of two variables in the same graphic. The resources (2) change the appearance according to its state, as seen in Figure 2, where two idle resources and a busy one are presented. On the other hand, activities (4) show only the number of process instances being processed.

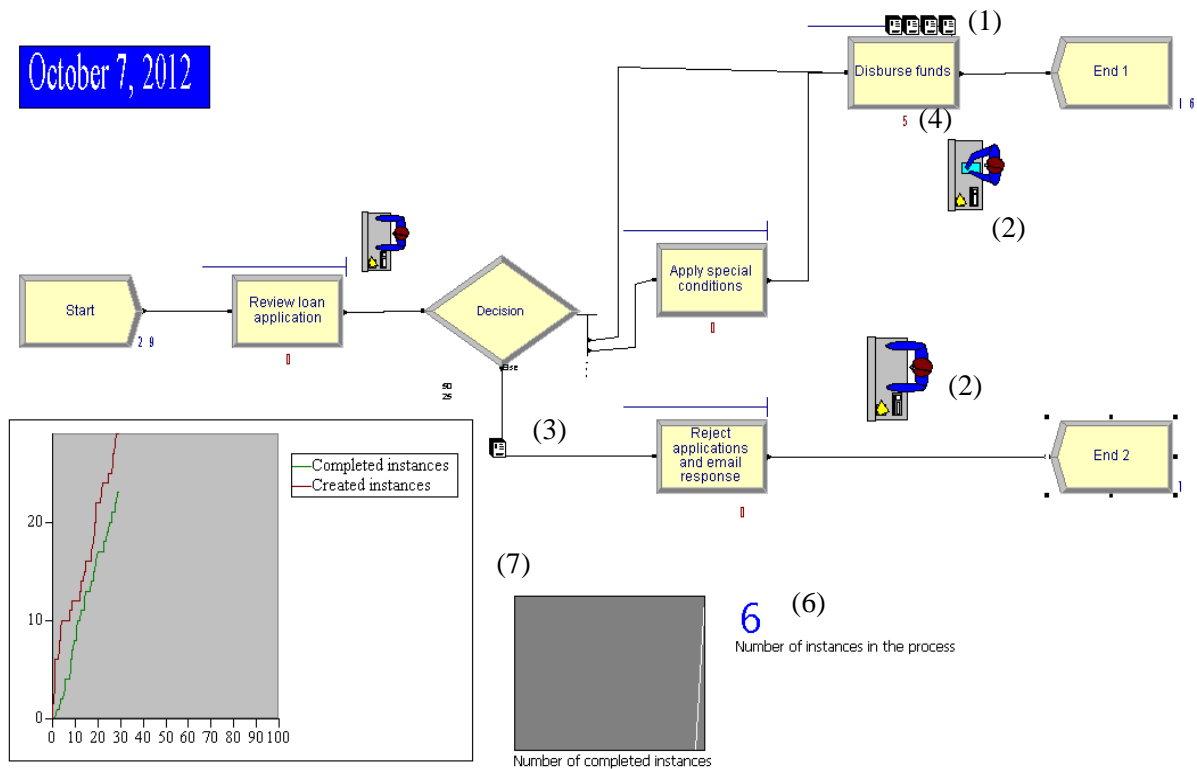


Figure 2 - Arena animation example

Customization is also provided in Arena, by allowing the user to choose the animated figure for all components, except for activities and sequence flows. It is also possible to customize graphics for attribute/global variable presentation. For the activities it is only possible to customize the numeric values that are presented, by choosing which activity attributes should be presented. The assessment outcome for this tool is displayed in Table 17.

Table 17 - ARENA evaluation scores

	Queue	Resource	Process instance	Activity	Sequence flow	Attributes and global variables
Component animation	4	4	3	2	1	4
Animation customization	4	4	3	2	1	4
	Speed controls		Interaction controls		Visualization controls	
Interaction controls	4		4		4	

3.3. Simul8

This is also a general purpose simulation tool. The evaluated version supports modeling with BPMN, but the animation capabilities were not as good as the ones that the tool provides for its proprietary modeling language, what lead us to choose the latter.

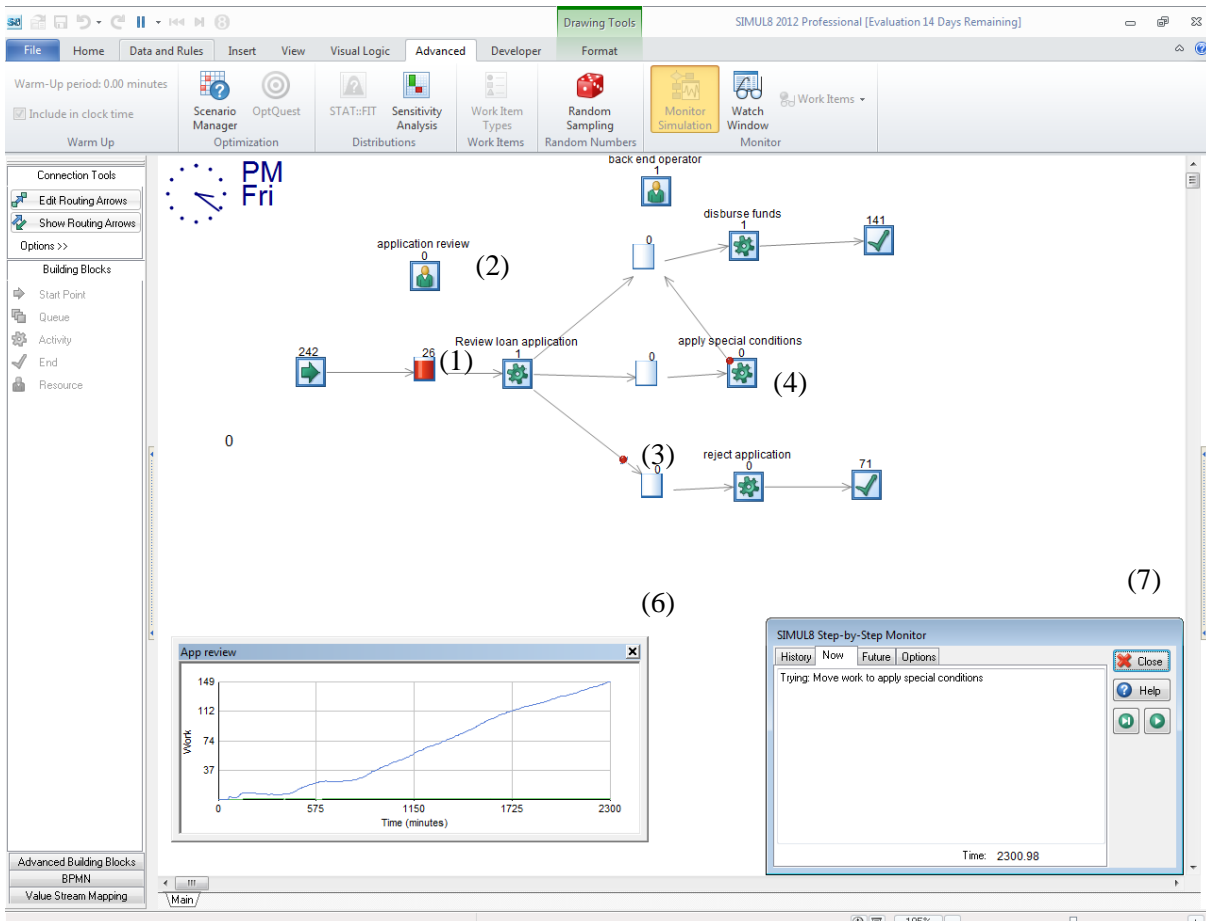


Figure 3 - Simul8 animation example

In Figure 3 the queues (1) are represented as a tank, together with a numeric value that shows the number of process instances waiting in the queue. The presentation used for resources is not for each

individual resource, but for a group of resources that is responsible for the activity. The resources (2) presentation does not change according to their state. Instead, are the activities (4) that have an associated state (e.g.: idle, busy or fail) and change its presentation according to it. It is possible to show the movements of resources from one activity to another when a resource is allocated to more than one activity. There is also a numeric value next to the resource figure that indicates the quantity of resources from that group that are not being used by an activity. Process instances can be programmed to change their appearance. In Figure 3 instances (3) appear as red dots. Simul8 supports the presentation of attributes as numeric values and also supports graphics (6) that can combine several attributes/global variables.

Simul8 provides animation customization capabilities, namely the “visual logic” feature allows the user to configure changes to an activity, a process instance and resources appearance. It is possible to choose from some available animations for the queue presentation. Regarding the resources, it is only possible to choose the representative static figure for each resource group, while process instances can have their figure changed over time. Simul8 allows customizing the activity figure for each state, as well as constructing graphics that can be composed by attributes and/or global variables. Concerning the interactivity controls, Simul8 provides a feature called “monitor simulation” (7) that enables the user to see the log of events, the event being processed and the future events that will be performed. This feature enables also the user to play an event at a time. Regarding the speed controls, Simul8 allows to play, pause and stop, choose the animation speed and jump the animation to a certain point in the simulation. The assessment outcome for this tool is displayed in Table 18.

Table 18 - Simul8 evaluation scores

	Queue	Resource	Process instance	Activity	Sequence flow	Attributes and global variables
Component animation	4	2	4	4	1	4
Animation customization	4	2	4	4	1	4
	Speed controls		Interaction controls		Visualization controls	
Interaction controls	4		1		1	

3.4. IBM WebSphere business modeler advanced 7.0

This tool supports BPMN and combines modeling, simulation and analysis features (IBM 2012). Figure 4 shows the queues animation (1) presented together with a numeric value, which indicates the number of instances waiting in the queue. The queue animated figure consists in a set of sticks that change color, one by one, as the number of process instances increases. When the queue becomes full, all sticks change to red. Process instances (3) are presented with the same figure (red marker). In

Figure 4 we can also see the activities animation (4) with the number of instances being processed and also the color change from green to grey, indicating that the activity is being executed. Sequence flow (5) turns red when an instance travels through it.

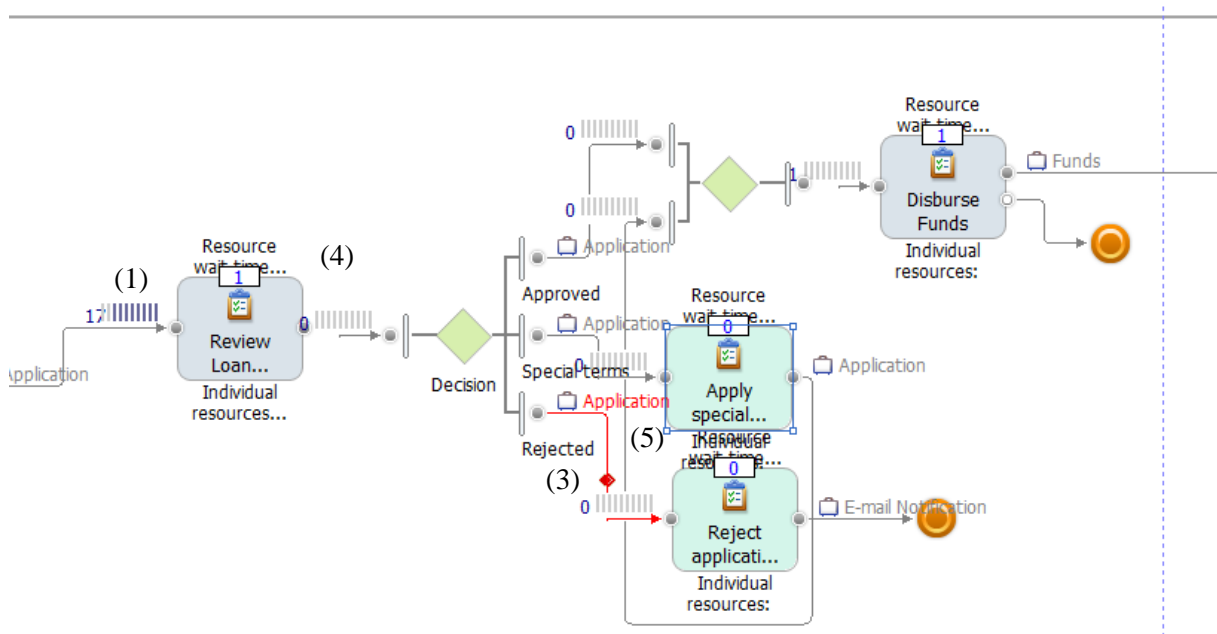


Figure 4 - Websphere animation example

Websphere does not support any type of customization for the animation, and regarding the interaction controls it features only start, pause and stop. The assessment outcome for this tool is displayed in Table 19.

Table 19 - Websphere evaluation score

	Queue	Resource	Process instance	Activity	Sequence flow	Attributes and global variables
Component animation	3	1	3	2	2	1
Animation customization	1	1	1	1	1	1
		Speed controls	Interaction controls	Visualization controls		
Animation interaction		2	1	1		

3.5. Progress savvion process modeler 8.0

This is a business process modeling tool that supports BPMN and has some simulation animation features (Progress 2012). Its market relevance is corroborated by Gartner (Sinur and Hill 2010). Although its animation features are not impressive, since it only covers activities and sequence flows, it has some aspects that deserve some attention, like its ability to identify high throughput activities, as well as possible bottlenecks (Progress 2012). As represented in Figure 5, an activity (4) turns red if the

total of its processed instances is greater than the average of all activities processed instances. It turns blue if that total is less than the average and white if the total is equal to the average. Besides, the total number of processed instances for an activity is shown. The sequence flow (5) follows the same color changing logic as the activities, but based on the number of process instances that travelled through the sequence flow.

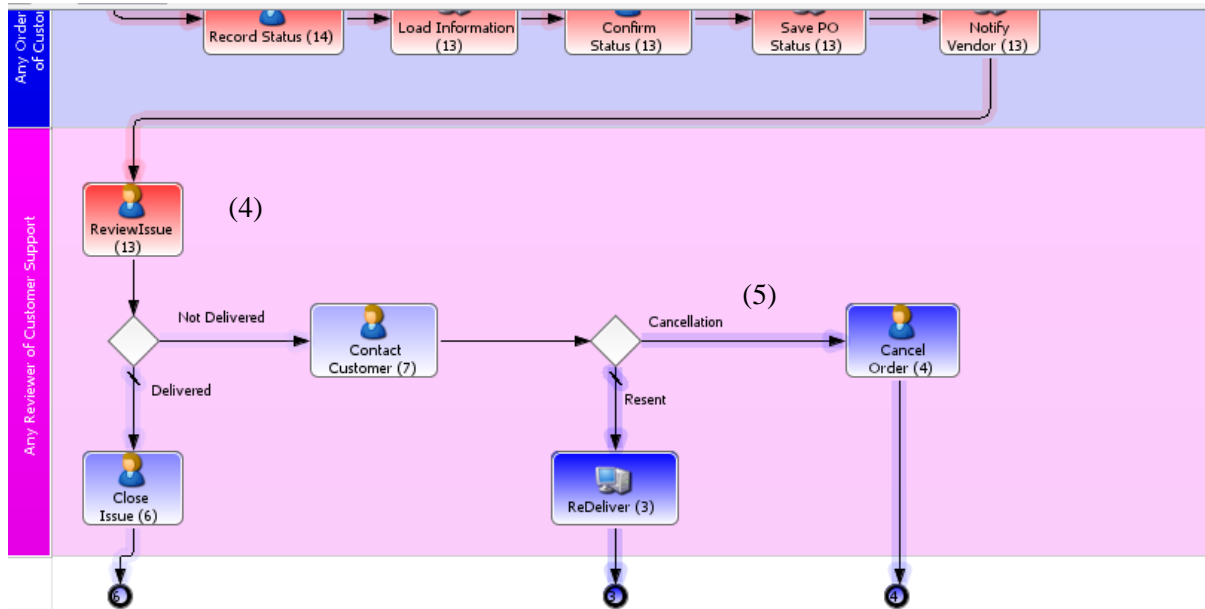


Figure 5 - Savvion animation example

Animation customization is not supported for any model component and the only interactivity controls that are provided are the speed control, start, pause and stop. The assessment outcome for this tool is displayed in Table 20.

Table 20 - Savvion evaluation score

	Queue	Resource	Process instance	Activity	Sequence flow	Attributes and global variables
Component animation	1	1	1	4	3	1
Animation customization	1	1	1	1	1	1
		Speed controls	Interaction controls	Visualization controls		
Animation interactivity		4	1	1		

3.6. Tibco business studio community edition

Tibco Business Studio is another example of a tool that provides business process modeling with BPMN and has simulation capabilities. Tibco is also a major player in this area, as identified by Gartner (Sinur and Hill 2010).

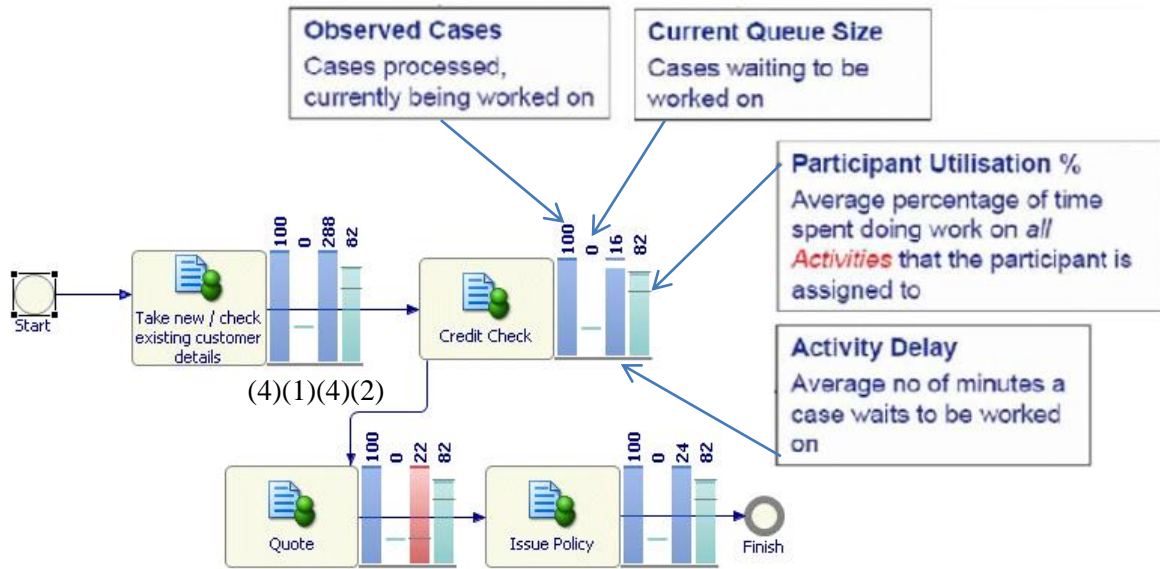


Figure 6 - Tibco animation example

Business Studio animation features include four bars placed at the right side of the corresponding activity. Those bars change colors from green to blue and finally to red. Each bar represents a different attribute and has a numeric value on top. The bar attributes are presented in Figure 6. The first and third bars represent the activity animation (4), the second represents the queue animation (1) and the fourth represents the resource animation (2). The process instances animation is not shown in any figure, but as the evaluation results suggest, it only indicates the path of the sequence flow through the presentation of several colored circles that pass through the sequence flow continuously.

The tool does not support any kind of animation customization and the interactivity controls provided are just the animation speed controls that allow controlling the animation speed, as well as to play, pause and stop the animation. The assessment outcome for this tool is displayed in Table 21.

Table 21 - Tibco evaluation score

	Queue	Resource	Process instance	Activity	Sequence flow	Attributes and global variables
Component animation	3	3	2	3	1	1
Animation customization	1	1	1	1	1	1
	Speed controls		Interaction controls		Visualization controls	
Interactivity controls	4		1		1	

3.7. ProM UITopia

ProM UITopia is the only process mining tool in our sample. Its creators claim that business process models can be seen as roadmaps and several cartography ideas (e.g. difference between highways and local roads representation in a map) can be used to increase models' understandability (Aalst et al. 2012). The roadmap metaphor contributes with animation innovations that are not present in the other analyzed tools.

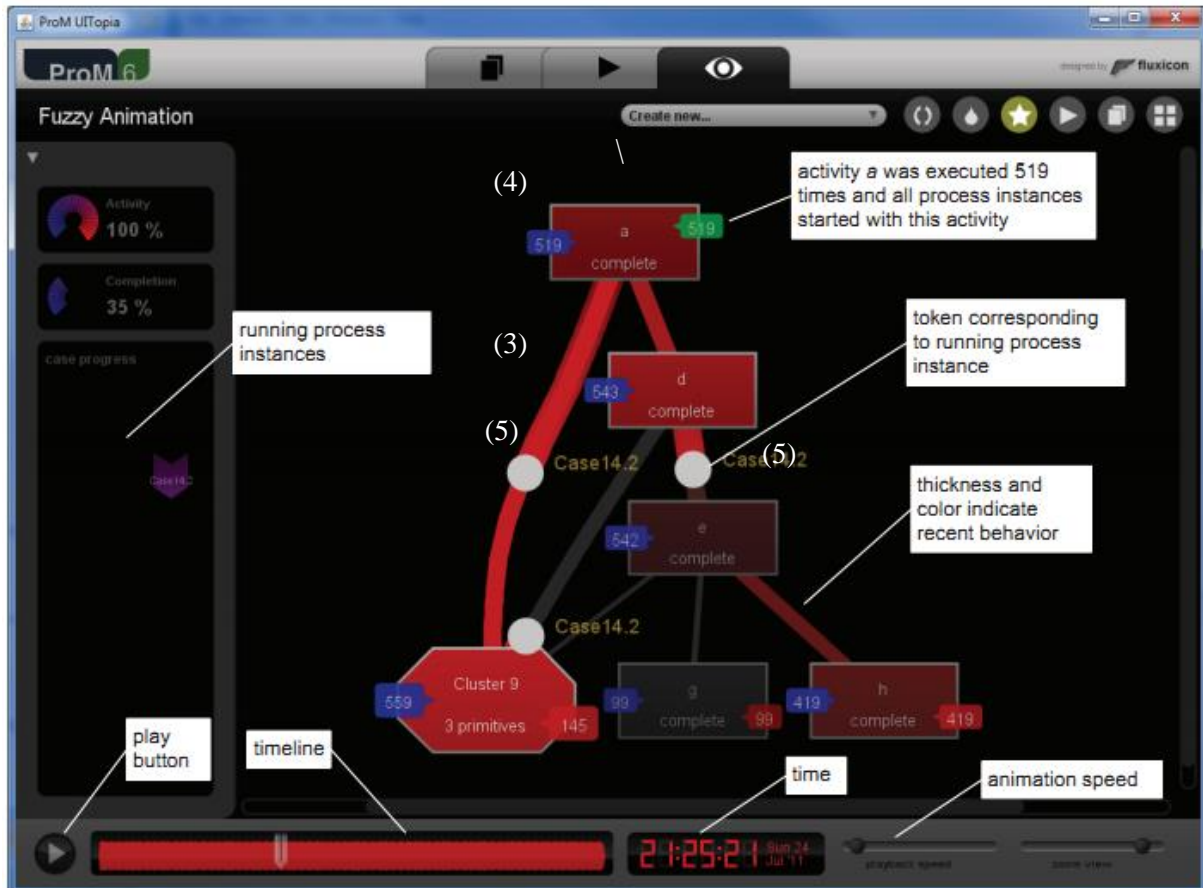


Figure 7 - ProM UITopia animation example (source: (Aalst, Leoni and Hofstede 2012))

ProM UITopia animation is compared with a movie that shows process execution traces. In ProM the latter were captured through process mining techniques, instead of being generated by a simulation. The implementation of the cartography metaphor is as follows: color brightness and thickness of sequence flows change during animation (5), depending on the number of instances that traversed them. Process instances (3) are presented with a white circle that leaves a trail as a comet. Activities (4) also change its color going from a darker red to a brighter one as its number of execution increases.

Animation customization is not possible for any model component and the available animation interactivity is just the animation speed, pause and play controls. The assessment outcome for this tool is displayed in Table 22.

Table 22 - ProM UITopia evaluation score

	Queue	Resource	Process instance	Activity	Sequence flow	Attributes and global variables
Component animation	1	1	3	4	4	1
Animation customization	1	1	1	1	1	1

	Speed Controls	Interaction controls	Visualization controls
Animation interactivity	4	1	2

3.8. Results summary

The summary of the assessment exercise for components animation is presented in Table 23. Notice that the full classification range (1 to 4) was observed for all component animation elements.

Table 23 - Summary of scores for component animation

	Queue	Resource	Process instance	Activity	Sequence Flow	Attributes / global vars
Arena	4	4	3	2	1	4
Simul8	4	2	4	4	1	4
Websphere	3	1	3	2	2	1
Savvion	1	1	1	4	3	1
Tibco	3	3	2	3	1	1
ProM UITopia	1	1	3	4	4	1

The summary of the assessment exercise for animation customization is presented in Table 24. The full classification range (1 to 4) was observed for all component animation elements, except for sequence flows.

Table 24 - Summary of scores for animation customization

	Queue	Resource	Process Instance	Activity	Sequence Flow	Attributes / global vars
Arena	4	4	3	2	1	4
Simul8	4	2	4	4	1	4
Websphere	1	1	1	1	1	1
Savvion	1	1	1	1	1	1
Tibco	1	1	1	1	1	1
ProM UITopia	1	1	1	1	1	1

The summary of the assessment exercise for interactivity controls is presented in Table 25.

Table 25 - Summary of scores for animation interactivity

	Speed controls	Interaction controls	Visualization controls
Arena	4	4	4
Simul8	4	3	1
Websphere	2	1	1
Savvion	4	1	1
Tibco	4	1	1
ProM UITopia	4	1	2

To conclude our tool survey we have produced an ordering of the sampled tools regarding their achieved values on the 15 scores proposed in our taxonomy. Figure 8 presents a multiple histogram where, from left to right (on each tool), we have a column for the frequency of score 4, 3, 2 and 1.

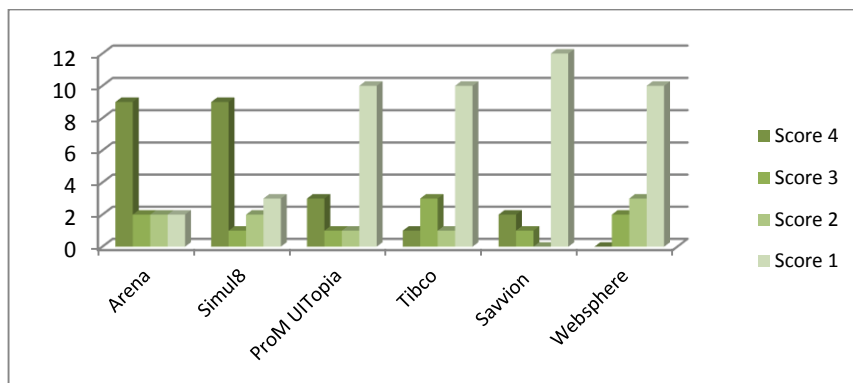


Figure 8 - Tool scores comparison histogram

On Table 26 we have calculated a final score for each tool that allowed us to rank them. That aggregated score is a weighted sum of the frequencies of each of the animation features. The weight in that sum is the value of each score. For instance, the final score of Arena was calculated as follows:

$$Score_{Arena} = 9 * 4 + 2 * 3 + 2 * 2 + 2 * 1 = 48$$

Table 26 - Final rank of surveyed tools regarding animation features

Tool	Score 4	Score 3	Score 2	Score 1	FINAL SCORE
Arena	9	2	2	2	48
Simul8	9	1	2	3	46
ProM UITopia	3	1	1	10	27
Tibco	1	3	1	10	25
Savvion	2	1	0	12	23
Websphere	0	2	3	10	22

Concluding, we have roughly two groups of tools when it comes to the support of process execution animation features. On one side we have the process simulation tools (Arena and Simul8) that have overall very sophisticated animation features. On the other side we have the process mining (ProM) and process modeling tools (Tibco, Savvion and Websphere). The latter lag behind considerably on animation and customization features for attributes and global variables, as well as on interaction and visualization controls.

3.9. Related work

To find out if related works exist, we searched on *IEEE Xplore*, *ACM Digital Library*, *SpringerLink*, *ScienceDirect* and *Wiley Online Library*, using the following search string:

"business process model" AND (simulation OR animation OR monitoring) AND "tool survey"

We were able to find very few related works. Only two of them are close enough to deserve being mentioned. In the first, seven criteria categories, each with a score, that can be used to evaluate and choose a simulation tool, are proposed (Tewoldeberhan 2002). Those criteria cover simulation, modeling, execution, testing and process animation aspects. However, animation features are not discussed in detail in this research work.

Another paper surveys business process simulation visualization aspects (Du et al. 2012). Although this work does not present classification criteria or tool evaluation, it categorizes the existent simulation animation features in three types: static graphic, dynamic animation and virtual reality.

3.10. Summary

This tool survey was particularly useful in providing us an updated view on the current state-of-the-art of business process models animation features.

Although the tool survey presented risks to be outdated in a few years' time, as new versions of the tools become available, we believe that our proposal for the classification of animation features in process execution will remain useful on a larger timeframe. As such, this survey can be seen as a twofold contribution:

- A feasibility study for the proposed taxonomy.
- A leverage for our research work regarding business process models animation.

Chapter 4

OSS tool selection

Contents

4.	OSS tool selection	34
4.1.	Introduction	34
4.2.	OSS tools evaluation framework.....	34
4.3.	Tools search and selection.....	36
4.4.	Description of the selected OSS projects	38
4.5.	Collected data from the OSS projects	40
4.6.	BPMN 2 elements coverage	41
4.7.	Fit for purpose evaluation criteria	42
4.8.	Applying fit for purpose evaluation criteria	45
4.9.	Results summary	54
4.10.	Summary	55

In this chapter we propose a framework for OSS tool selection together with its application in a sample of OSS tools gathered from several web-based software source code repositories.

4. OSS tool selection

4.1. Introduction

One of our expected contributions is a fully functional prototype that implements BPMN model animation features. Our plan is then extending an existing open source BPMN modeling tool that meets the following basic requisites:

- provide good modeling capabilities (preferably compliant with BPMN 2);
- be developed in Java;
- be successful as an open source project;
- be understandable (for extendibility sake).

We searched for related work on OSS tools evaluation frameworks to find a framework that we could use to understand which of the available OSS BPMN modelers available we should use to extend. Since we were not able to find related work besides the ones that we found regarding OSS projects success, we decided to develop and propose an OSS tools evaluation framework. It can be used to assess OSS project success and fit for purpose.

4.2. OSS tools evaluation framework

To assess the success of each OSS project, we propose, herein, a set of five criteria and corresponding indicators. This assessment framework, summarized in Table 27, is partly inspired in the proposals contained in (Crowston et al. 2003; Crowston et al. 2004), as indicated.

Table 27 - OSS projects success evaluation criteria

Project activity <ul style="list-style-type: none"> o Number of releases o Average release period (Crowston, Annabi, Howison and Masango 2003) o Number of project members (Crowston, Annabi, Howison and Masango 2003) o Project creation date o Last update date 	Project Nature <ul style="list-style-type: none"> o Tool type
User evaluation <ul style="list-style-type: none"> o Number of downloads (Crowston, Annabi, Howison and Masango 2003) o User ratings o Number of evaluations 	Available documentation <ul style="list-style-type: none"> o API o Tutorials o User Guide/Manual
Forum activity <ul style="list-style-type: none"> o Number of posts o Number of topics o Registered users 	

We will now provide details on each criterion and corresponding indicators.

Project activity – In order to evolve, a project needs to have a certain level of activity. Higher levels of activity lead to more frequent code releases, so it is important to analyze the projects activity to see if they are currently active and progressing. In order to assess the activity, the following indicators are proposed:

Number of releases – Indicates the number of new versions that were released; a higher number of releases represents a more successful project; this indicator should be analyzed together with the “average release period” and the “project creation date” to grant comparability across projects.

Average release period – Indicates the average time between releases; the liveliness of the projects’ development process is reverse proportional to that period of time.

Number of project members – Indicates the number of contributors to the project; since many OSS projects are based on volunteer contributors, the capability to attract them can be considered as a success indicator.

Project creation date – Indicates the date when the project was created; an older date can be considered a better one, since it shows that the project has been capable to maintain its activity for a longer period of time.

Last update date – Indicates the project last update date; a recent date indicates that the project is still active and evolving.

User evaluation – This criterion relates to the users’ perception of the OSS tool. The following indicators are proposed for user evaluation:

Number of downloads – This number is easy to get in most cases; however, it can be an unreliable indicator, since not all downloads result in actual usage.

User ratings - Indicates how the users rated the tool, based on collected data from the available project information in the repository site; this indicator is not available for all tools, so it should be analyzed together with the “number of evaluations” indicator.

Number of evaluations – Corresponds to the number of times a tool was evaluated.

Forum activity – Forums dedicated to tools are a good source of information on how to use them and also on how to understand its code. The community that gathers around a forum exchanges knowledge and helps newcomers to get insight of the project more quickly. So, a project with a larger community would be preferable. The following indicators are proposed for forum activity:

Number of posts – This is the total number of posts that exist in the forum.

Number of topics – This is the total number of topics that exist in the forum.

Registered users – This is the total of number of users registered in the forum; although not all of the registered users participate in the forum topics, this indicator can be used as a reference for the community size.

Project Nature – This criterion identifies the type of tool that is being evaluated. This can be useful to understand which type is more appropriate, according to the evaluators experience and knowledge in certain technologies. The following indicator is proposed for this criterion:

Tool type – This is a nominal scale indicator that identifies the type of tool being evaluated. Possible values include (but are not restricted to): Eclipse IDE plug-in, Web Platform, and Rich Client Application.

Available documentation – This criterion assesses the availability of documentation about tool usage and development. The following indicators are proposed for this criterion:

API – This predicate indicates if there is information available about the tool API.

Tutorials – This predicate indicates if tutorials about the tool usage are available (e.g. in Youtube).

User Guide/Manual – This predicate indicates if a manual or guide exists to facilitate tool usage.

4.3. Tools search and selection

4.3.1. Search criteria

We started by surveying the available open source projects to find the one that best serves our requirements. The search was performed in six open source project repositories: Sourceforge²; GitHub³; Google code⁴; Codeplex⁵; Javaforge⁶ and Gitorious⁷. In each of these repositories we used the

² **SourceForge**: Is a web-based source code repository for software development projects – <http://sourceforge.net/>

³ **GitHub**: Is a web-based hosting service for software development projects that use the Git revision control system – <https://github.com/>

⁴ **Google Code**: Is Google's site for developer tools, APIs, technical resources and project hosting – <http://code.google.com/>

⁵ **CodePlex**: Is an open source project hosting website from Microsoft – <http://www.codeplex.com/>

⁶ **JavaForge**: Is an open source software development community with a hosting portal for open source projects – <http://www.javaforge.com>

following search string: “BPMN AND (Animation OR Simulation OR Modeling OR Modeler)”. The number of results for each repository is presented in Table 28.

Table 28 - Number of results by open source projects repository

	Number of results
Sourceforge	7
GitHub	88
Google Code	12
Codeplex	0
Javaforge	0
Gitorious	0

4.3.2. Search results

To trim down the number of results presented in Table 28 we started by checking the relevance of those projects. After eliminating irrelevant projects, we analyzed the remaining ones using the evaluation criteria presented in Table 27 and we eliminated projects that were not active or that did not have a forum available. The final list of projects to be further analyzed is present in Table 29. The web addresses for the homepage, source code repository and forum of each OSS project are presented in Table 60.

⁷ Gitorious: Is a web-based hosting service for collaborative free and open-source software development projects that use the Git revision controlsystem – <http://gitorious.org/>

Table 29 - OSS projects found in the search process

Project name	Repository
o Bonita Open Solution: Open Source BPM	
o jBPM	o Sourceforge
o Modelio – Modeling environment (UML)	
o Yaoqiang BPMN Editor	
o Activiti	o GitHub
o Bpmn2 Modeler	

4.4. Description of the selected OSS projects

4.4.1. Activiti

Activiti is a process engine for Java (Activiti 2013) that enables the modeling and execution of business processes using an Eclipse plugin to model and create webforms for the human activities. *Activiti* also provides a web application for processes administration, although that is not taken in account for our work.

4.4.2. Bonita open solution: Open source BPM

Bonita is a combination of modeling and execution tool. It enables the user to model processes using BPMN and to develop the web forms for each activity. User can execute the model with the help of a workflow (Bonitasoft 2013). Besides these features, *Bonita* also provides process diagrams validation and simulation capabilities.

The studio, which is used to model and simulate the processes, is a Rich Client Platform developed in Java and the sources are available in a SVN repository.

4.4.3. Bpmn2 Modeler

Bpmn2 Modeler is an Eclipse plugin being developed in Java, integrated in the SOA Platform Project⁸. It is a modeling tool that allows the creation and edition of BPMN diagrams. *Bpmn2 Modeler* is built

⁸ SOA: Is a software design and software architecture design pattern based on discrete pieces of software providing application functionality as services to other applications – <http://www.eclipse.org/projects/project.php?id=soa>

upon Eclipse Graphiti⁹ and uses the Bpmn2 EMF Meta model¹⁰ which is compatible with BPMN2 OMG specification.

4.4.4. jBPM

jBPM is a light-weight, extensible, workflow engine that enables business processes execution modeled in BPMN2 (JBoss 2013). It provides the possibility to model complex business logic by combining business rules and complex event processing in the modeled business processes. *jBPM* is developed in Java and it offers two suites for process modeling, one web-based designer and an Eclipse plugin.

4.4.5. Modelio – Modeling environment (UML)

Modelio presents itself as “a modeling environment, supporting a wide range of models and diagrams, and providing model assistance and consistency checking features” (ModelioSoft 2013). This tool “combines BPMN support and UML support in one tool with dedicated diagrams to support business process modeling” (ModelioSoft 2013), besides these standards *Modelio* also supports TOGAF, XMI and SoaML among others. *Modelio* uses the RCP framework¹¹ from the Eclipse open source platform to manage the GUI structure of the tool, and the GEF¹² Eclipse graphical library to support *Modelio* diagrams (ModelioSoft 2013). *Modelio* is developed mostly in Java except for a small part developed in C++ which is entitled as core. For extensibility purposes, *Modelio* offers a scripting language support (*Jython*)¹³ and the capability of receiving “modules” that can be perceived as plug-ins. These modules can be developed with the help of examples and documentation available in the *Modelio* website. They can be developed to interact with Modelio API in order to handle the models, get needed information and manipulate them.

4.4.6. Yaoqiang Bpmn editor

Yaoqiang is a modeling tool that allows the user to create, view, edit and simulate business processes (Yaoqiang 2013). It is dedicated to BPMN2 and supports Import/Export of OMG BPMN2 files. The tool functions as a rich client application and does not need installation.

⁹ Eclipse Graphiti: Eclipse-based graphics framework that enables rapid development of state-of-the-art diagram editors for domain models – <http://www.eclipse.org/graphiti/>

¹⁰ BPMN2 EMF meta model: metamodel implementation based on the Business Process Model and Notation (BPMN) 2.0 OMG specification – <http://www.eclipse.org/modeling/mdt/?project=bpmn2>

¹¹ RCP: The minimal set of plug-ins needed to build a rich client application – http://wiki.eclipse.org/Rich_Client_Platform

¹² GEF: Is a framework that was developed for the Eclipse platform – <http://www.eclipse.org/gef/>

¹³ Jython: Is a successor of JPython, an implementation of the Python programming language written in Java – <http://www.jython.org/>

There is not much information about the technical aspects of the tool, only that it is all developed in Java and that it has an Extensible Plugin Architecture although we could not find documentation or examples that explain how to use this feature.

4.5. Collected data from the OSS projects

Table 30 - Selected OSS projects data

Modelio	Yaoqiang BPMN Editor	Bonita Open Solution	JBPM	Activiti	Bpmn2 Modeler
---------	----------------------	----------------------	------	----------	---------------

Project activity

Number of releases	4	162	13	48	19	N/A
Average releases period	84 days	6 days	39 days	73 days	52 days	N/A
Number of Project Members	13	1	3	198	146	6
Project creation date	01-02-2012	26-05-2010	20-08-2009	02-01-2003	28-02-2012	01-08-2011
Last update date	08-10-2012	14-03-2013	13-03-2013	16-11-2012	03-03-2013	04-03-2013

Users evaluation

Number of Downloads	2.164	24.900	9.279	1.458.303	N/A	N/A
User Ratings	100%	89%	59%	N/A	N/A	N/A
Number of evaluations	12	140	142	13	N/A	9

Forum activity

Number of posts	1.530	N/A	41.290	N/A	2.584	311
Number of topics	309	112	9.186	1.842	552	82
Registered users	595	N/A	56.389	768	3.350	N/A

Project Nature

Project type	RCA	RCA	RCA / Web Platform	Eclipse Plug-in	Eclipse Plug-in	Eclipse Plug-in
--------------	-----	-----	--------------------	-----------------	-----------------	-----------------

Documentation

API available	Yes	Yes	Yes	Yes	Yes	Yes
Tutorials available	Yes	No	Yes	No	Yes	Yes
Guide / Manual	Yes	No	Yes	Yes	Yes	No

N/A – Not Available

The data collected to assess the OSS projects success is presented in Table 30. In terms of project activity we conclude that all projects are still active. Yaoqiang is the project with the higher number of releases followed by jBPM. Regarding the average number of releases it is also Yaoqiang the project

that presents the best value followed by Bonita Open Solution. The project that presents the larger number of project contributors is jBPM followed by Activity. We were not able to get the number of releases for the BPMN2 Modeler project. However, we were able to see that the project had, at the time of the analysis, frequent commits in its source code repository. In term of project activity we conclude that all projects are still active.

In terms of users evaluation it is possible to see that jBPM has the greater number of downloads followed by Yaoqiang.

Regarding forum activity it seems that all projects have an active community. However we can conclude that Bonita Open Solutions has the larger community.

All projects have at least one type of documentation available, and the majority of the tools have two types of documentation available.

We can, then, conclude that all projects in our sample are active and successful. We can also conclude that all projects presented in the sample should be further analyzed regarding their BPMN2 elements coverage in order to see if they should be analyzed in our fit for purpose analysis.

4.6. BPMN 2 elements coverage

A BPMN 2 modeling tool is expected to have a high coverage of the notation that includes 86 graphical elements. We established a minimum coverage of 50% (43 elements) for tool acceptance.

Table 31 - BPMN 2 coverage results

	Activiti	Bonita	BPMN2-Modeler	jBPM	Modelio	Yaoqiang BPMN
Total number of BPMN elements	31%	38%	53%	10%	69%	77%
	(27)	(33)	(46)	(9)	(59)	(66)

In Table 31 we can observe a large diversity in notation coverage. Tools like Activiti, Bonita and JBPM that focus on the process execution have a smaller palette of BPMN 2 elements comparing to the other selected tools. The two tools that have the highest coverage are Modelio and Yaoqiang BPMN with 69% and 77%, of coverage, respectively. Taking into account our minimum of 50% of BPMN 2 elements coverage, Activiti, Bonita and jBPM were excluded from our Fit for purpose evaluation.

4.7. Fit for purpose evaluation criteria

The objective of this evaluation is to realize which of the three remaining projects is the most suitable for our needs. We analyzed the projects in terms of source code accessibility, API extensibility, source code understandability and general complexity of the project source code.

4.7.1. Source code access

The access to some projects source code is very easy because their owners reveal the link for the repository where the project source code is available in read-only mode. On the other hand there are projects which source code is harder to get because it is not revealed directly in the project repository site. Instead the source code is available for download in zip format from the project official site. This criterion tries to evaluate the easiness of access to project source code. We were especially interested in the location of a link that would give us access to the source code through a configuration management system (e.g. SVN, GIT or Mercurial).

Table 32- Source code access easiness

<p>Unavailable</p> <p>The source code is not accessible.</p>
<p>Hard</p> <p>The source code is not available directly from a repository. It is only available for download in zip format.</p>
<p>Medium</p> <p>The source code is located in a repository in read-only mode, but the link for the repository is hard to find or the one given needs changes.</p>
<p>Easy</p> <p>The source code is located in a repository in read-only mode and the link for it is presented in the project repository site.</p>

4.7.2. API flexibility

The existence of a flexible application programming interface is important in order to add new functionalities to a project without making direct changes to its source code. This criterion aims to measure the projects API flexibility regarding a list of required functionalities for the project to be

developed. An example of required functionalities can be seen in section 4.8.2. The API flexibility can be classified by following the criteria presented in Table 33.

Table 33 - API flexibility classification criteria

<p>Inexistent</p> <p>The project API does not provide any of the required functionalities.</p>
<p>Not flexible</p> <p>The project API provides only half (50%) of the required functionalities.</p>
<p>Flexible</p> <p>The project API provides a great part (75%) of the required functionalities.</p>
<p>Very flexible</p> <p>The project API provides almost all (>75%) of the required functionalities.</p>

4.7.3. Source code understandability

This topic concerns two evaluation criteria: the number of languages in the project and documentation availability regarding Javadoc comments for classes and methods. For the first criterion a higher number represents a worst result, whereas for the second is the opposite.

Number of programming languages used

Just counting the number of different programming languages used may be a naive simplification. Several aspects should be considered which are related to that choice such as the syntactic and semantic complexity of each language, the available IDE support or the previous knowledge of the development team, to name a few (Dojo 2010). We believe a weighted metric sum would be required here, but we leave its discussion for future work.

Javadoc coverage

Javadoc can be used to enrich the Java code with information that can help developers to better understand it. The Javadoc coverage criterion evaluates if all classes and methods from the project have Javadoc comments and can be classified as presented in Table 34. In order to measure the Javadoc coverage we developed an Eclipse IDE plug-in that analyzes all the methods and classes of the project and calculates the percentage of classes and methods having Javadoc comments.

Table 34 - Javadoc coverage evaluation criterion

<p>Very Poor</p> <p>The Javadoc documentation covers a small part (less than 25%) of the methods.</p>
<p>Poor</p> <p>The Javadoc documentation covers less than 50% of the methods.</p>
<p>Good</p> <p>The Javadoc documentation covers between 50% and 75% of the methods.</p>
<p>Very Good</p> <p>The Javadoc documentation covers more than 75% of the methods.</p>

4.7.4. Source code complexity

Software engineering researchers have pointed out long ago that code complexity hampers its understandability (Lanning and Khoshgoftaar 1994). Understanding complex source code is more time consuming, making simpler source code a better choice. A metrics based analysis can be done to facilitate this choice. In section 4.8.4 we present an example of a source code complexity analysis for three OSS tools. In order to score the tools regarding code complexity, less complex tools are scored with lower score whereas more complex tools are scored with a higher score. When two tools present the same complexity in one criterion they should receive the same classification.

4.8. Applying fit for purpose evaluation criteria

4.8.1. Source code access results

Table 35 - Source code access results

	Bpmn2 Modeler	Modelio	Yaoqiang
Source code access	Easy	Hard	Easy

Both BPMN2 Modeler and Yaoqiang have their code available in a web repository in read-only mode and the link for the repository is available in the webpage of both projects. This justifies why both tools source code access is classified as “Easy”. On the other hand, Modelio was classified as having a “Hard” source code access. This means that the source code is not available directly from a repository, being just available for download in zip format.

4.8.2. API flexibility

In order to measure the API flexibility it is necessary to build a list of specific requirements related to the project that will be developed. For our study we built the requirements list presented in Table 36.

Table 36 - API requirements list

Model elements manipulation	Graphic elements manipulation
Add element to model	Add elements to the palette
Get activity associations (incoming and outgoing)	Add graphic element to diagram
Get activity type	Get model graphic elements
Get model elements	Get node location
Get model graphic element by model element	Get node size
Get resource roles	Set border line color
Get resources from resource role	Set diagram enabled/disabled
	Set link color
	Set link width
	Set node border width
	Set node color
	Set node location
	Set node size

Table 37 - API flexibility results

	BPMN2 Modeler	Modelio	Yaoqiang
API flexibility	Very flexible (100%)	Very flexible (95%)	Very flexible (100%)

Our list of requirements specifies a list of 20 features that we would like to find in a tool. The list is divided in two categories: model elements and graphical elements. The first refers to functionalities regarding BPMN model elements manipulation (e.g. getting the model elements existent in a diagram), while the second refers to functionalities regarding the manipulation of their graphical presentation. The results presented in Table 37 shows that all analyzed tools present a very flexible API regarding our requirements list.

4.8.3. Source code understandability

Number of Source code languages

Table 38 - Number of source code languages results

	BPMN2 Modeler	Modelio	Yaoqiang
Source code languages	1 (Java)	2 (Java and C++)	1 (Java)

Javadoc coverage

Table 39 - Javadoc coverage results

	BPMN2 Modeler	Modelio	Yaoqiang
Javadoc Method Coverage	Poor (33,35%)	Poor (29,89%)	Poor (45,77%)
Javadoc Class Coverage	Very Good (94,81)	Very Good (75,31)	Very Good (95,90)

4.8.4. Source code complexity

This complexity analysis is based on several tools source code. In order to facilitate this comparison, a multiple view environment for software visualization called SourceMiner¹⁴ was used.

Methods size analysis

Good object-oriented programming practices promote short methods (Fowler and Beck 1999). In order to compare the three tools, we use a treemap view that displays information about package, types and methods. The visualization is filtered to color only methods with more than 20 lines, a size threshold suggested in (Martin 2009).

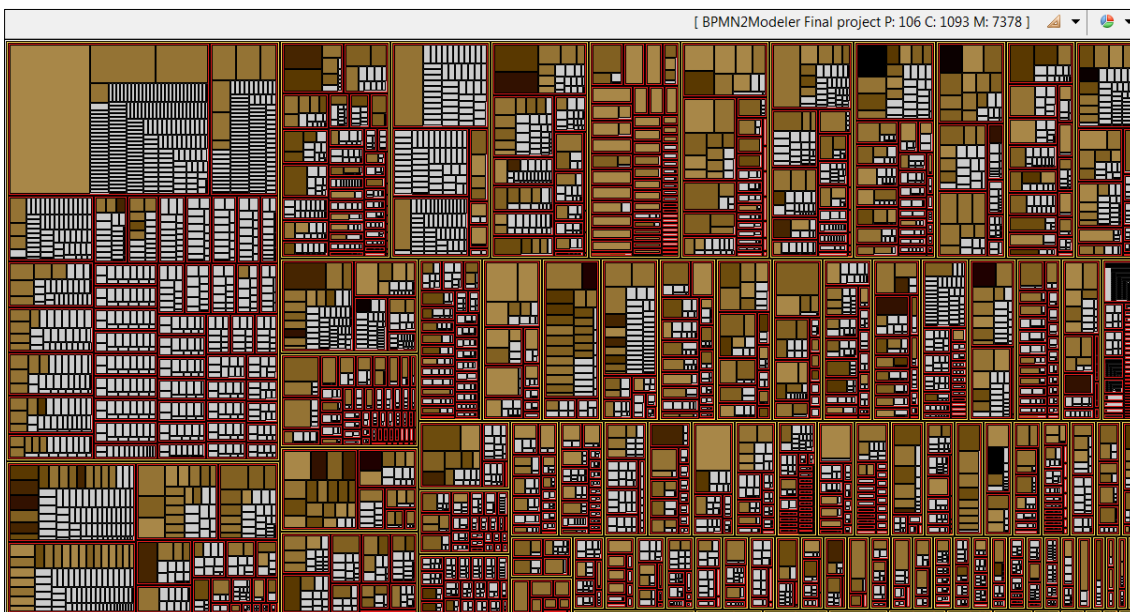


Figure 9 - BPMN2 Modeler size analysis treemap view

¹⁴ <http://www.sourceminer.org/> (accessed on December 2013)

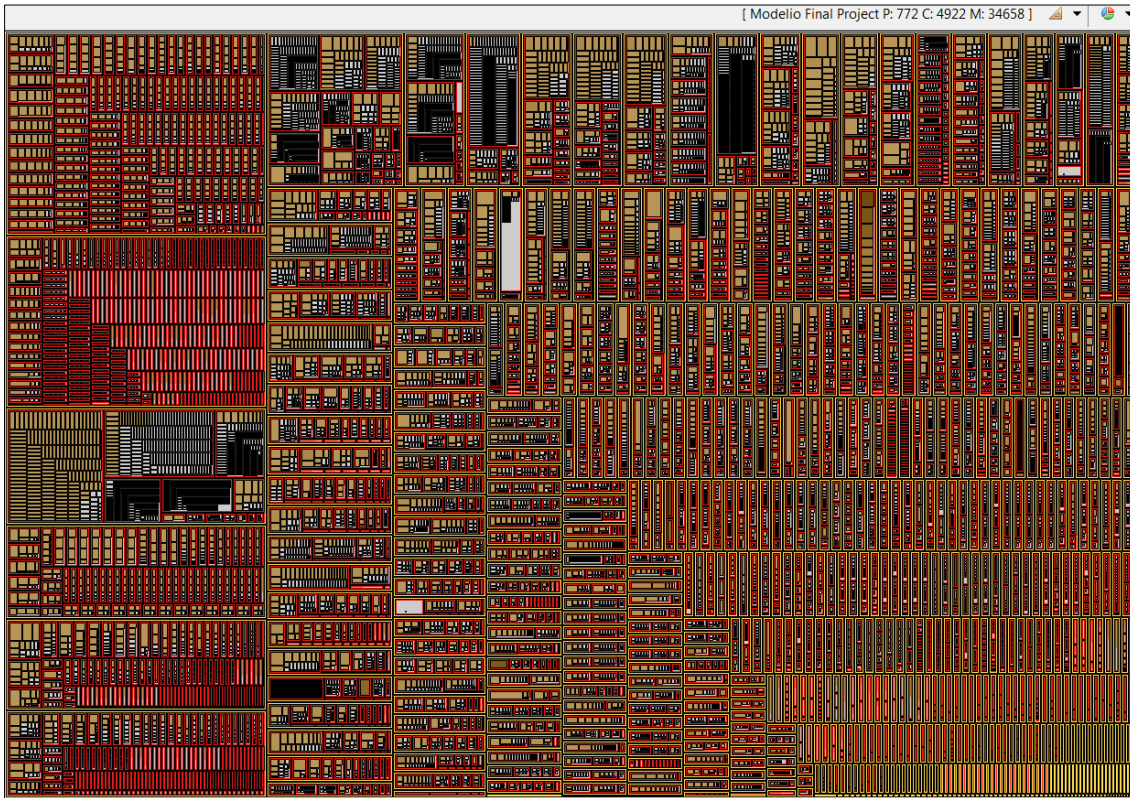


Figure 10 - Modelio size analysis treemap view

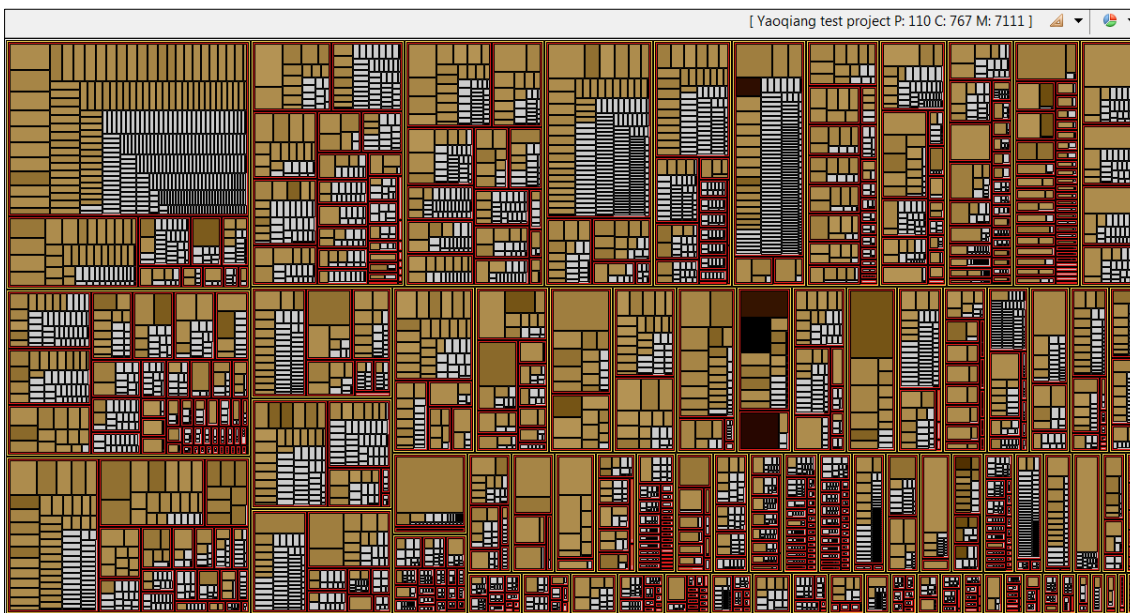


Figure 11 - Yaoqiang size analysis treemap view

Figure 9, Figure 10 and Figure 11 present the size analysis treemap views. They present the Packages limited by a yellow border, the types limited by a red border and finally the methods limited by a black border. The area of the rectangles shows the number of lines whereas the color represents the

complexity (darker color represents higher complexity). Modelio is the biggest of the three. It has a higher number of packages, types and methods which can be seen by comparing the aforementioned. BPMN2 Modeler and Yaoqiang are more similar. Nevertheless, the first has a smaller number of methods with more than 20 lines of code, which can be seen by comparing Figure 9 with Figure 11. This observation makes the BPMN2 Modeler the less complex in terms of the criterion under consideration.

Complexity analysis

Complex software modules are more difficult to test and maintain. Thus a tool with less complex modules is preferable. Again we have chosen a treemap view that presents the Package, Types, Methods hierarchy. In order to make the visualization easier to perceive the view only colors methods with complexity higher than 10, since it is considered by McCabe as a “reasonable, but not magical, upper limit” (McCabe 1976).

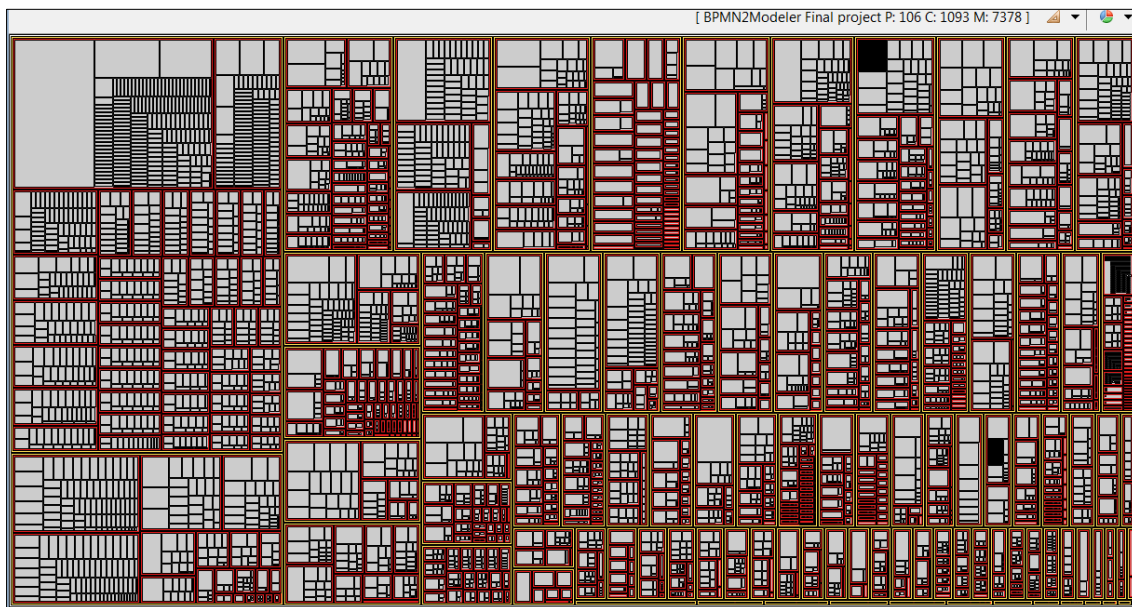


Figure 12 - BPMN2 Modeler complexity analysis view

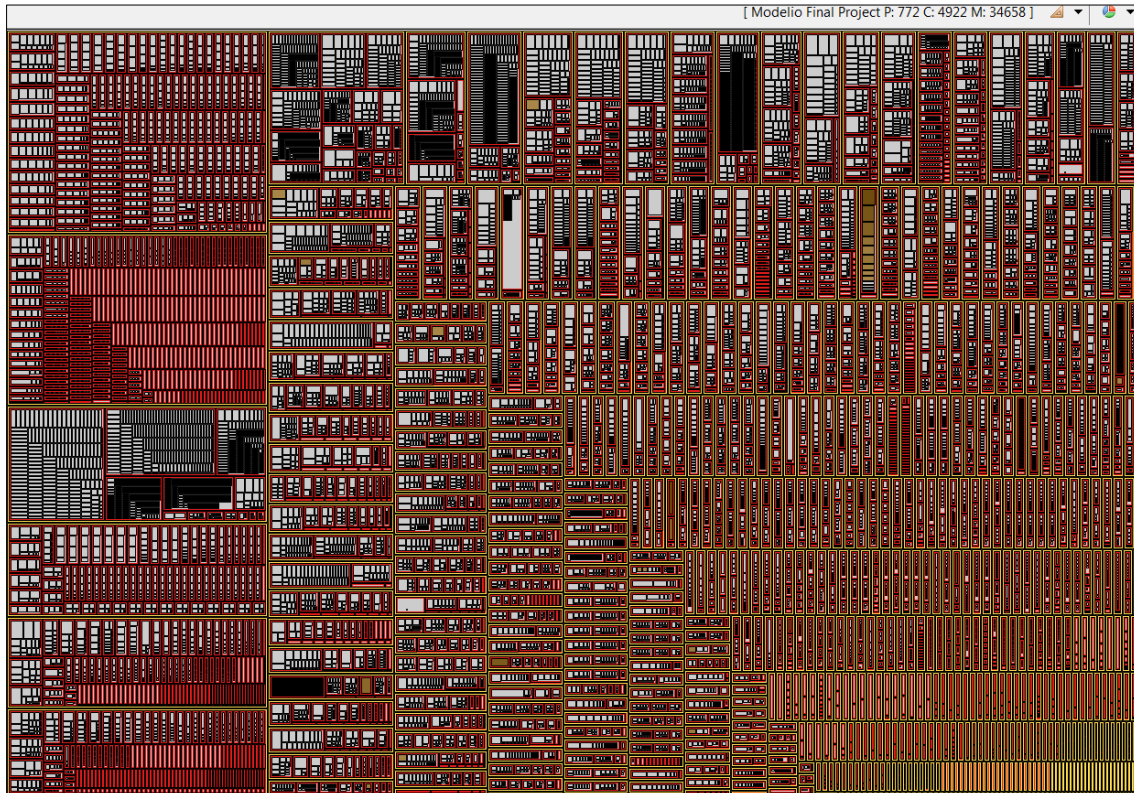


Figure 13 - Modelio complexity analysis view



Figure 14 - Yaoqiang complexity analysis view

The treemaps presented in Figure 12, Figure 13 and Figure 14 have the same display logic described in the Method size analysis topic. Modelio stands out from the three by being the one with more complex

regions, which are presented in black. Comparing Figure 12 with Figure 14 it is possible to notice that the second figure (Yaoqiang) shows a higher number of complex methods than Bpmn2 Modeler.

Coupling analysis

Coupling regards the existing dependencies between programming elements (e.g. packages, classes or methods). Complexity increases when coupling increases which means that a high value of coupling represents a less understandable solution (Pressman 2010). Two types of coupling are analyzed in this point: Afferent Coupling (CA) and Efferent Coupling (CE) as defined in (Martin 2003). The first refers to classes that are referenced by others, whereas the second refers to classes that depend on other classes. The Afferent Coupling visualization is filtered to show only classes that are in the bad threshold (CA greater than 20) for software that contains a number of classes between 101 and 1000, as proposed in (Ferreira et al. 2012). The Efferent Coupling visualization is filtered to show only classes with a CE greater than 9. This values is based on the Magical number 7 ± 2 limitation theory that explains the limitation of our working memory to store more than 9 chunks of information (Miller 1956). In this case, the chunks of information are the classes referenced by the one being analyzed, because in order to fully understand one class, it is required to understand the ones that it depends on.

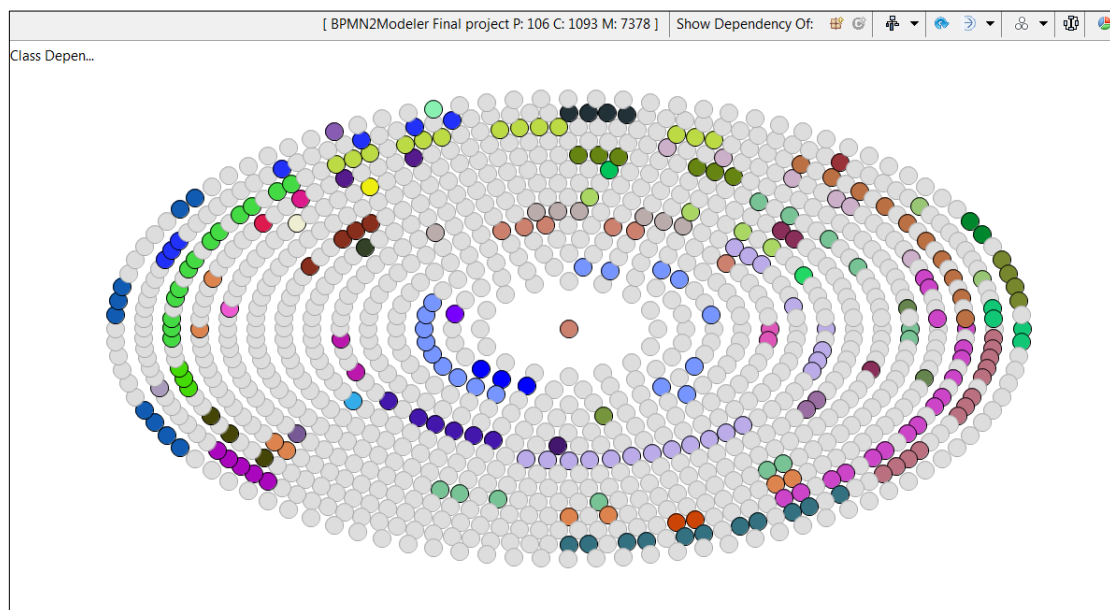


Figure 15 - Bpmn2 Modeler CA analysis with dependency view

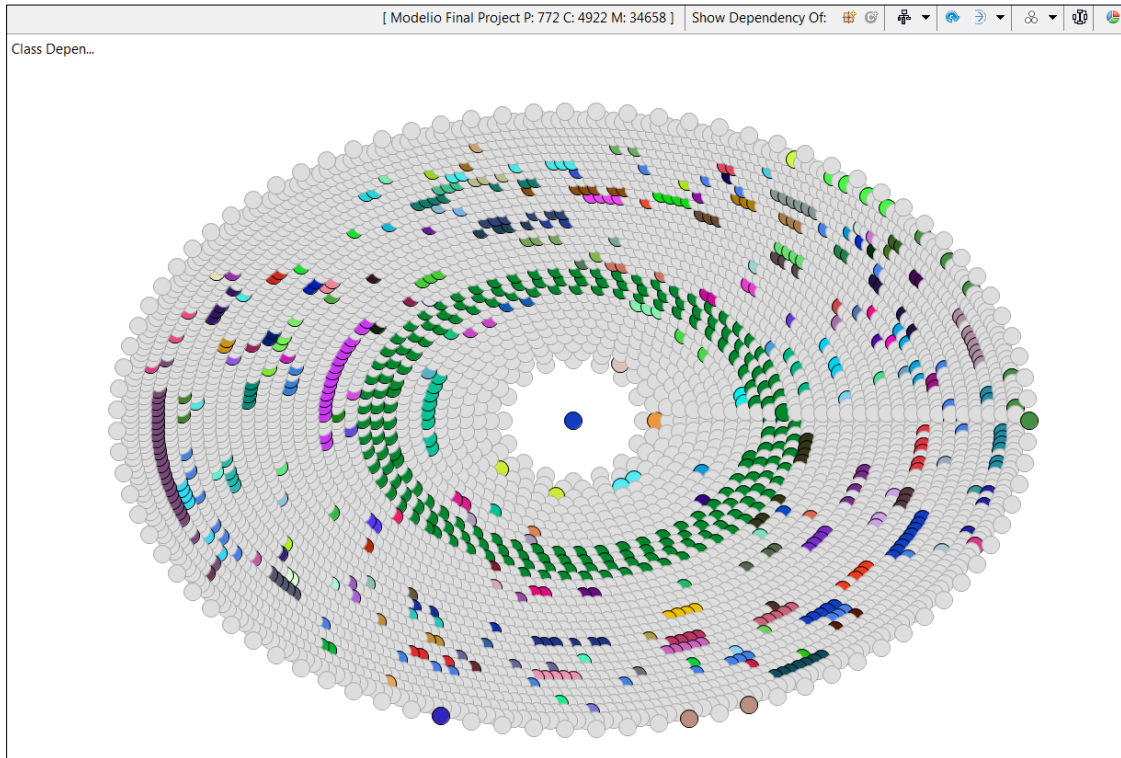


Figure 16 - Modelio CA analysis with dependency view

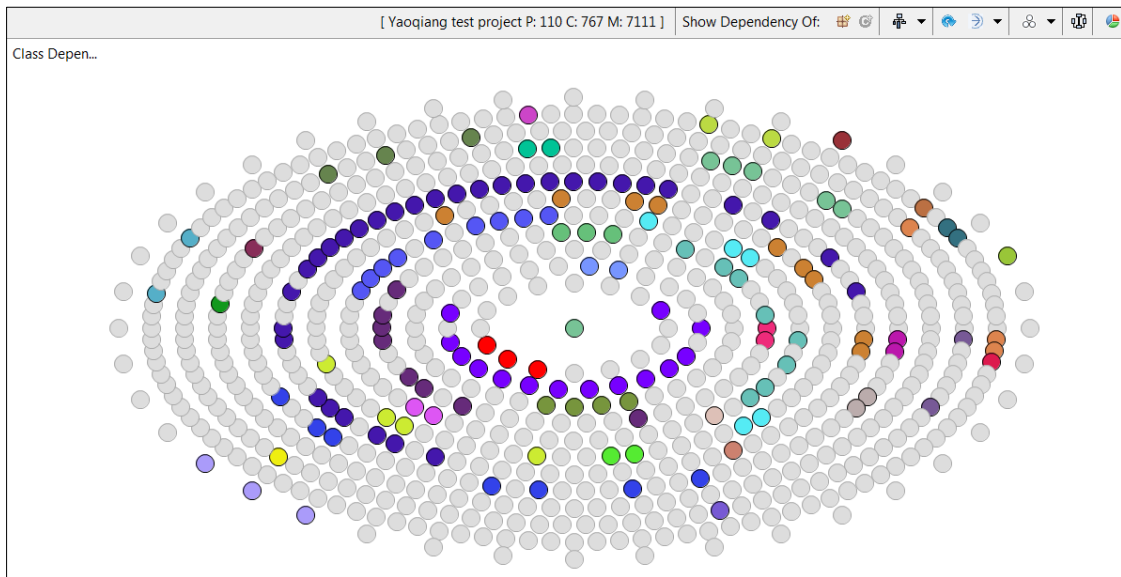


Figure 17 - Yaoqiang CA analysis with dependency view

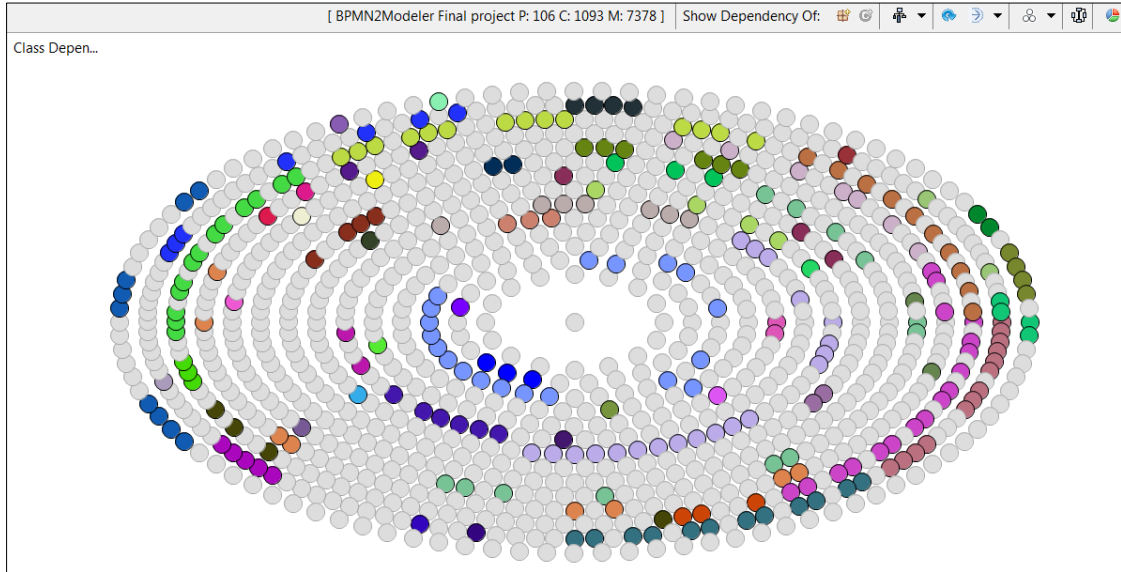


Figure 18 - BPMN2 CE analysis with dependency view

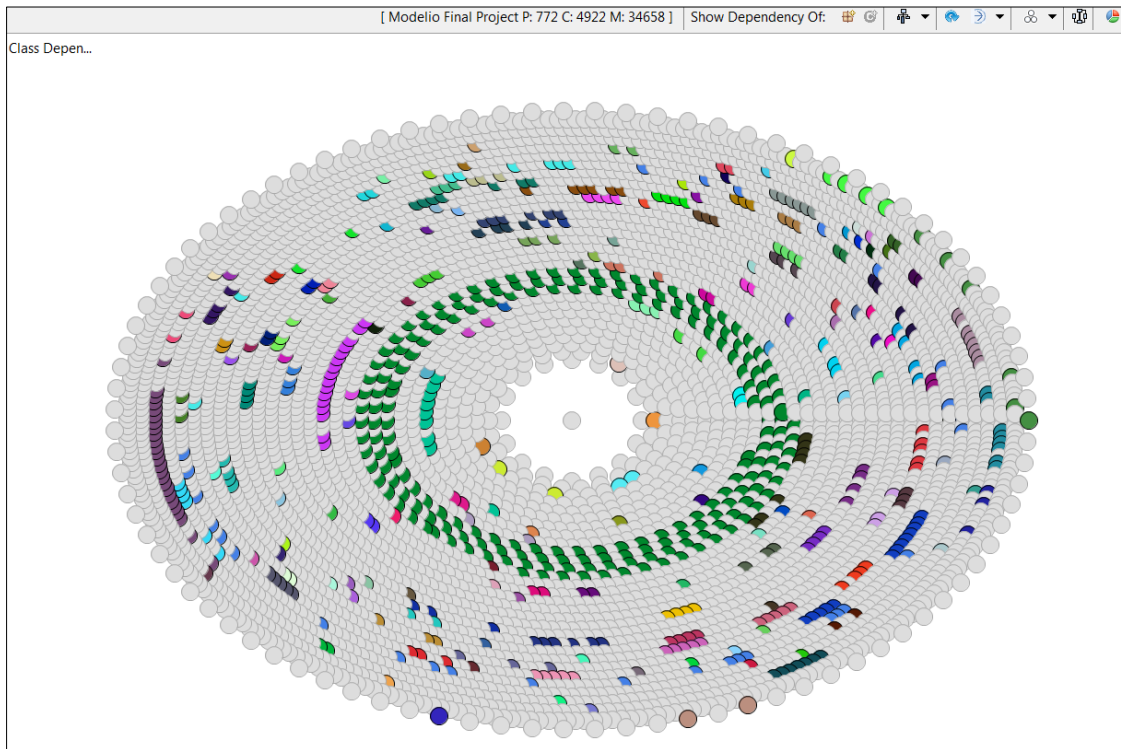


Figure 19 - Modelio CE analysis with dependency view

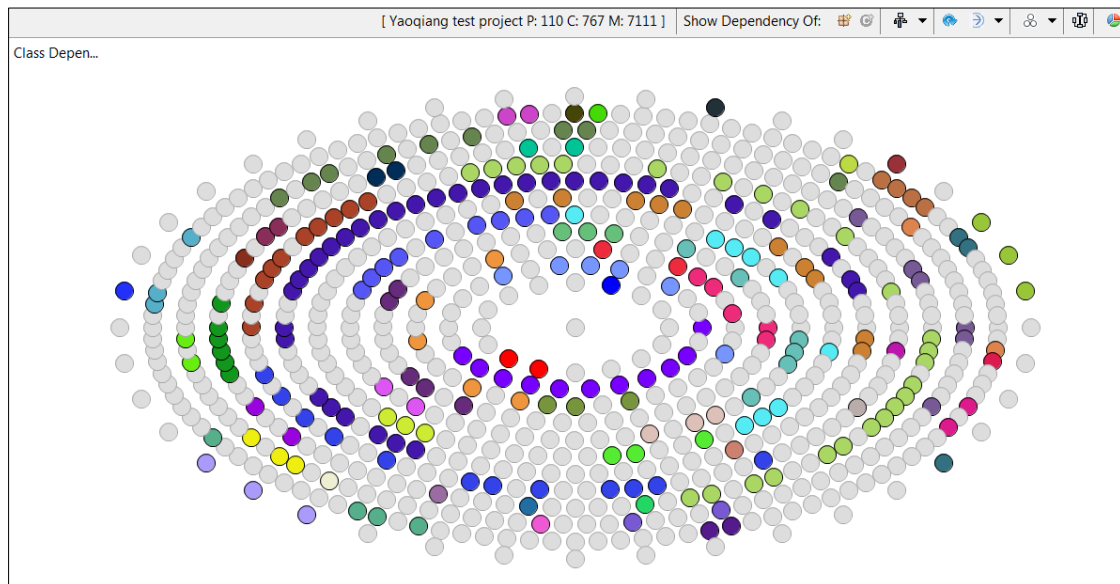


Figure 20 - Yaoqiang CE analysis with dependency view

Figures from Figure 15 to Figure 20 show a dependency view where each colored ball represents a class that has a value higher than the one set in the filter ($CA > 20$ and $CE > 9$). Both in CA and CE analysis Modelio has more colored classes than Yaoqiang or Bpmn2 Modeler, which shows, once again, its higher complexity. The comparison in terms of CE analysis between Bpmn2 Modeler and Yaoqiang is not conclusive, since it is not easily perceived, from viewing both Figure 20 and Figure 18, which presents the higher number of colored classes. However, through the comparison of Figure 15 with Figure 17, that presents the CA analysis, it is possible to notice that Yaoqiang presents less colored classes.

4.9. Results summary

A results summary of the performed fir for purpose evaluation is presented in

Table 40. Using these results we can conclude that Modelio is the less desirable OSS tool to use, regarding our stated. The results show that Modelio source code is harder to get than the other two projects source code. It uses 2 source code languages while the other tools use only 1, and it has the worst source code complexity score in all analyses, which makes it the most complex tool of the three. Bpmn2 Modeler and Yaoqiang are very similar and therefore have a very similar maintainability effort score that only differs by one point. The existent difference between the two is regarded with the source code complexity: Bpmn2 Modeler received a better classification than Yaoqiang in size and complexity, whereas Yaoqiang received a better classification in Afferent coupling analysis. Based on our results we can conclude that Bpmn2 Modeler seems to be the best choice from the three tools regarding our requirements and the criteria on our assessment framework.

Table 40 - OSS tools analysis summary

	BPMN2 Modeler	Modelio	Yaoqiang
Source code access	Easy (1)	Hard (3)	Easy (1)
API flexibility	Very flexible (1)	Very flexible (1)	Very flexible (1)
	100%	95%	100%
Source code understandability			
Source code languages	1	2	1
	(Java)	(Java and C++)	(Java)
Javadoc method coverage	Poor (3)	Poor (3)	Poor (3)
	33,35%	29,89%	45,77%
Javadoc class coverage	Very Good (1)	Very Good (1)	Very Good (1)
	94,81	75,31	95,9
Source code analysis			
Size analysis	1	3	2
Complexity analysis	1	3	2
Afferent Coupling analysis (CA)	2	3	1
Efferent Coupling analysis (CE)	2	3	2
Maintainability effort score	13	22	14

4.10. Summary

In this chapter we introduced a framework to facilitate the analysis and choice of OSS tools, which is composed by two components. One regards the search and selection of relevant tools based on project activity, forum activity and documents availability. The other regards a fit for purpose analysis used to calculate a maintainability effort score of each tool.

Table 40 shows the final result of the fit for purpose analysis, where it is possible to see that the tool with the lower maintainability effort score, and therefore, the better result is the BPMN2Modeler, which means that it will be extended to implement the BPMN2 animation concepts presented in the next chapter.

[This page is intentionally blank]

Chapter 5

BPMN 2.0 elements animation

Contents

5.	BPMN 2.0 elements animation.....	58
5.1.	Introduction	58
5.2.	BPMN 2.0 elements	58
5.3.	BPMN 2.0 proposed new elements	59
5.4.	Animation for BPMN 2.0 and proposed elements	63
5.5.	Animation views.....	67
5.6.	BPMN 2.0 elements graphical rules	69
5.7.	Summary	70

In this chapter BPMN 2.0 elements are presented followed by a presentation of new proposed elements that will be used during process models animation. Then, the animation proposed ideas are presented for both BPMN 2.0 and new proposed elements. Animation views are also presented and finally the BPMN 2.0 elements graphical rules are analyzed to check violations by our animation proposed ideas.

5. BPMN 2.0 elements animation

5.1. Introduction

In this chapter BPMN elements are presented. Because BPMN does not have a corresponding element for every modeling component identified, new graphical elements are proposed. An animation for each modeling element as well as animation views are proposed herein. The objective with this work is to animate the BPMN Process sub-model type. Animation for elements of the Choreography and Collaboration types are outside the scope of this proposal.

5.2. BPMN 2.0 elements





The specification of BPMN identifies five basic categories of elements, which are: Flow Objects; Data; Connecting Objects; Swimlanes and Artifacts.

From this list, the first category is the one that contains the most used elements, which are: Events; Activities; and Gateways.

Another important category is the Connecting Objects. This category contains the following elements: Sequence Flows; Message Flows; Associations; and Data Associations.

All of these elements have a graphical element in the specification as displayed in Table 41.

Table 41 - BPMN 2.0 Basic elements notation (source: (OMG 2011))

Element	Notation
Events	
Activities	
Gateways	
Sequence Flow	

Message Flow	
Association	

BPMN does not support graphically Queues, Resources or Process Instances. Therefore, to present a process execution through animation using the BPMN notation it is necessary to extend it with these missing graphical elements.

5.3. BPMN 2.0 proposed new elements

5.3.1. Resources

The notion of resource exists in BPMN 2.0. The notation specification indicates that the Resource class should be used to indicate the resources referenced by Activities. Figure 21 shows an extract of the BPMN metamodel related with resource assignment.

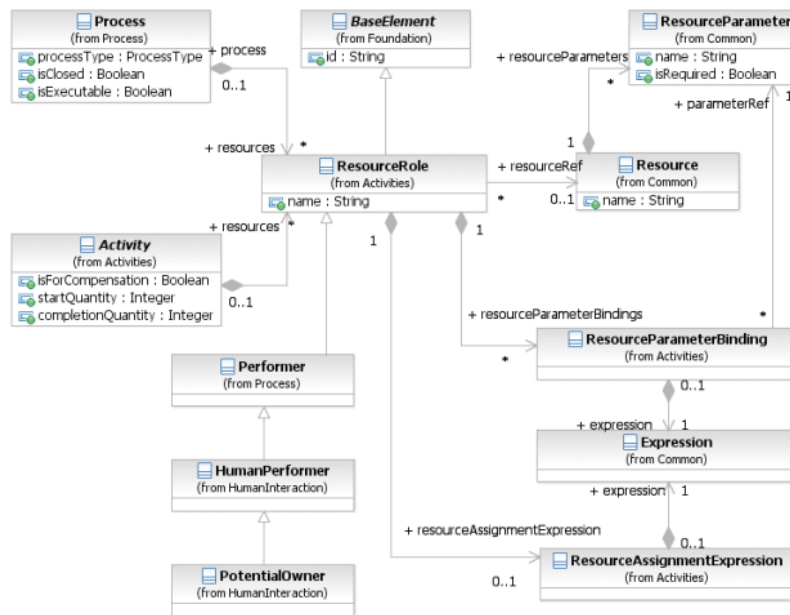


Figure 21 - BPMN metamodel extract related with resource assignment (source: (OMG 2011)).

In Figure 21 is possible to see that a resource is assigned to an activity through a resource role. However, neither the resource, nor the resource role, has a graphical notation, so the elements in the Figure 22 and Figure 23 are proposed to represent the resources roles in BPMN models.



Figure 22 - Graphical notation proposed for human resource roles



Figure 23 - Graphical notation proposed for automated resource roles

A resource role that represents humans should be presented with the icon presented in Figure 22, whereas a resource role that represents automated resources, for example computers or other tools, should be presented with the icon displayed in the Figure 23.

Table 42 - Resource graphical notation used by BPMN task type









	Resource image
Task	
Send Task	
Receive Task	
User Task	
Manual Task	
Business Rule Task	
Service Task	
Script Task	
Sub-process	-

Table 42 shows which resource role icon should be displayed for each task type. The table shows that the human resource role icon is only used for tasks performed by humans: user task and manual task; whereas the automated resource role icon can be used in the rest of the activities.

5.3.2. Queues

As seen in Figure 24, BPMN 2.0 does not encompass the notion of queue in its metamodel. However, this is an essential component in diagrams that are used for simulation or monitoring of business processes. In order to extend BPMN 2.0 with the notion of queue, a new metaclass Queue was added

as presented in Figure 25. For the graphical presentation of queues, a small rectangle positioned on top of the activities can be used, like in the example presented in Figure 26.

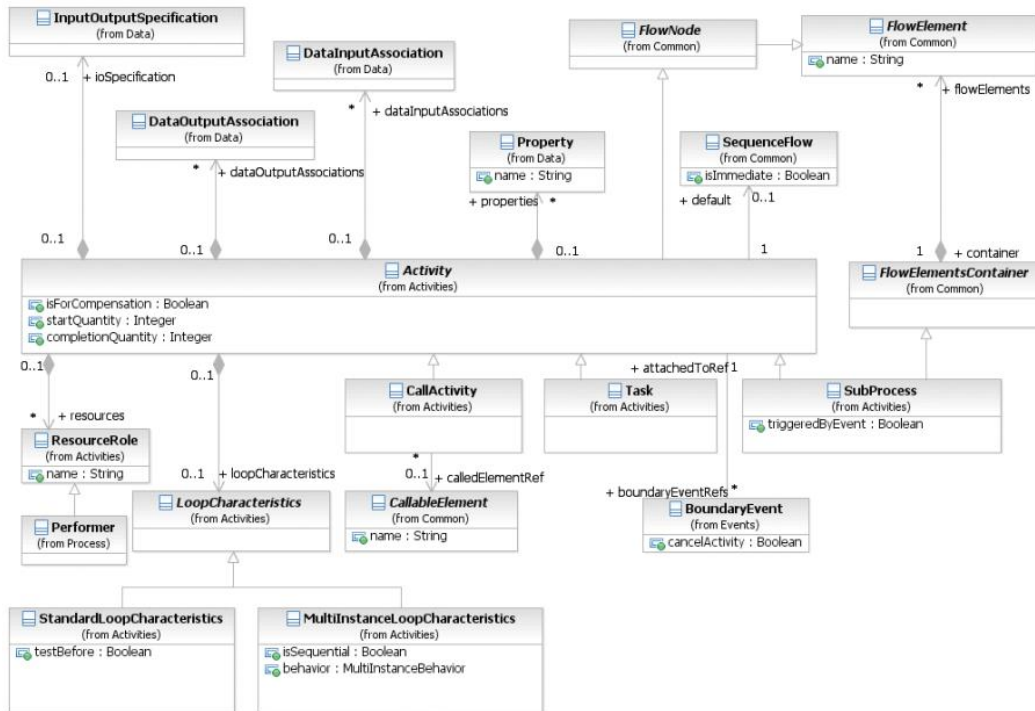


Figure 24 - BPMN metamodel extract related to activities (source: (OMG 2011)).

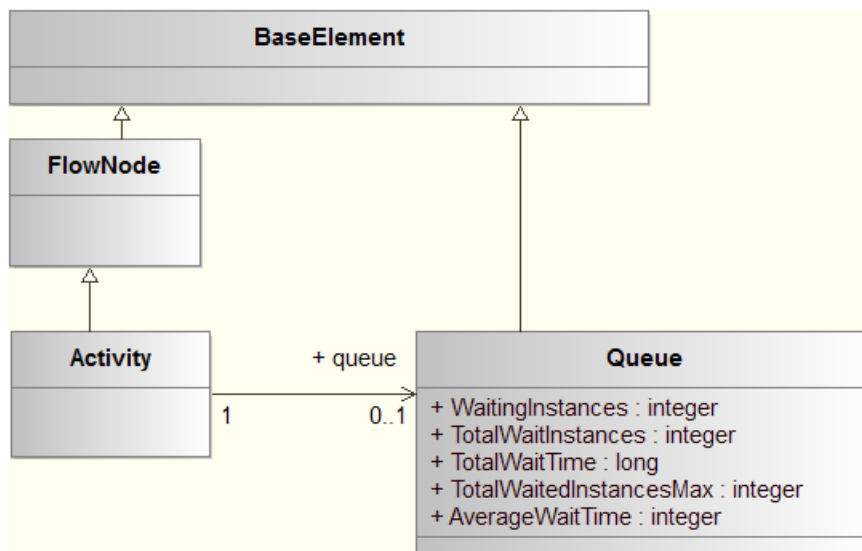


Figure 25 - Propose extension to the BPMN metamodel

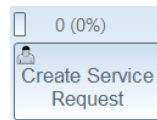


Figure 26 - Proposed graphical notation for a Queue.

5.3.3. Process instances

During the execution of the process, new instances are created and processed in the activities that compose the model. They can be seen as tokens that travel through the workflow. Similarly to the resources, the BPMN 2.0 notation introduces the concept of process instance, but does not support it graphically. With the intent of enriching the BPMN 2.0 notation with the graphical notation of instance we first need to extend the BPMN metamodel, as represented in Figure 27 where the instance metaclass is proposed as a subclass of *BaseElement*.

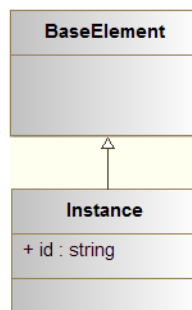


Figure 27 - Instance metaclass

Regarding graphical notation, the proposed default representation for process instances is a simple blue dot that travels the process sequence flows. The instance travel may be represented in two ways. The first is “Move around”, where the instance is presented traveling from the beginning of a sequence flow to its end. The second way is “Appear only once”, where the instance appears only in the middle of the sequence flow for a short time and then disappears. Process instances can take other, more expressive graphical representations to convey more information about a process instance. We now present some proposals for alternative process instance icons.



Figure 28 - Icon for human process instances.

Human instance icon – can represent clients, patients or other roles that can be played by a human actor.



Figure 29 - Icon for document process instances.

Document instance icon – can represent any type of document like, for example an invoice or a purchase order.



Figure 30 - Icon for default process instances.

Simple instance icon – represents a generic type of process instance without any other meaning.

Other icons can be added to express further process semantics.

5.4. Animation for BPMN 2.0 and proposed elements

To show process execution information through animation, we need to inflict graphical changes to the model. We will now propose an animation for the notation elements and also for the new proposed elements.

In essence, there are three graphical characteristics that can be manipulated in the elements, which are: the border line thickness, the border line color (Figure 31), and the fill color of the element (Figure 32).

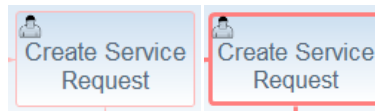


Figure 31 - Border line thickness and border line color change example

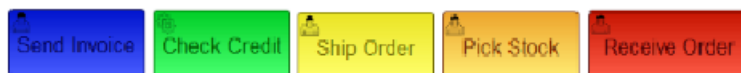


Figure 32 – Fill color change example

5.4.1. Fill color and border line color animation

The fill color of a BPMN element can be one of the following five: blue, green, yellow, orange and red. The colors evolve from blue to red just as we can see in Figure 33, which presents an example of the color spectrum. Such color spectrum is used in temperature presentation, for example, in thermal images where blue represents lower temperatures and red represents higher temperatures. Using the analogy of thermal images for BPMN elements animation means that elements filled with blue color present low values for the animated performance metric; whereas elements filled with red color have high values for the same metric.



Figure 33 – Color spectrum example

The color attribution for the BPMN animated elements should be done based on the comparison of the performance metrics values that each element has. When using fill color, the elements of the same type should be animated based on the same metric in order to ensure comparability between the animated elements. Concluding, the element with the lower value for the animated performance metric should be filled with the blue color whereas the element with the higher value should be filled with the red color. The other colors should be used having as reference the registered maximum for the performance metric. For example, an element which performance metric value is half of the maximum for that same metric should be filled with the yellow color. The same applies to the border line colors of the BPMN elements.

5.4.2. Border line thickness animation

The animated border lines evolve from thin to thick. When an element is presented with a thin line, it means that the element has a low value for the performance metric that is being animated with the border line thickness. On the other hand, a thick line means that the element has a high value for the metric that is being animated with the border line thickness. Similar to the fill color animation, the animation of the border line thickness should have as reference the maximum value registered for the performance metric.

5.4.3. Events animation

BPMN 2.0 defines the following types of events: Start, intermediate (throw and catch), boundary and end event.

Start and End Event

BPMN 2.0 specification defines specific rules for the thickness of the border line of Start and End events. For Start Event it is defined that circle must be drawn with a single thin line, and that the thickness of the line must remain thin so that the Start Event can be distinguished from the Intermediate and End Events (OMG 2011). For the End Event it is defined that the circle must be drawn with a single thick line, and that the thickness of the line must remain thick so that the End Event can be distinguished from the Intermediate and Start Events (OMG 2011).

In order to respect these rules we only propose the border line color and the fill color animation for both Start and End Event.



Figure 34 – Start Event border line color and fill color animation example



Figure 35 - End Event border line color and fill color animation example

Intermediate Event

The presentation rules for the Intermediate Event indicate that it must be drawn with a double thin line. The thickness of the line must remain thin and it must remain a double line (OMG 2011). In order to respect the specification presentation rules we propose only the border line color and fill color animation. In the case of the border line color, only the outside circle border line is manipulated. The result of an animated event should be similar to the one presented in Figure 36.



Figure 36 - Intermediate Event border line color, and fill color animation

5.4.4. Activities animation

BPMN 2.0 notation identifies three types of activities: task (service task, send task, receive task, user task, manual task, business rule task, script task), call activity, sub process.

All of these activity types, except subprocess and call activity, can be animated with all types of animation proposed: border line thickness, border line color and the element fill color. The final result should be similar to what is presented in Figure 37.



Figure 37 - Examples of activity animation (*in the left: only border line thickness and border line color. In the middle: only fill color. On the right: combination of the three animation types*)

Call activities should only be animated with the element fill color. Border line color and thickness cannot be manipulated since the specification indicates that the activity must remain with a thick border line in order to ensure differentiation from the other tasks types.

5.4.5. Gateways animation

For the gateways we propose all three types of animation: border line thickness, border line color and fill color animation. The animation final result should be similar to Figure 38.

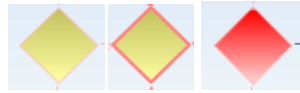


Figure 38 - Gateway border line thickness, border line color and fill color animation

5.4.6. Sequence flows animation

The notation identifies three types of sequence flow: sequence flow, default Sequence flow and conditional sequence flow.

For all of them, only border line thickness and color is appropriate, since sequence flows are nothing more than a line with an arrow on the tip. Thus, a variant of that animation type should be used, a variant that can be cataloged as line thickness and line color. The animation result should be similar to the one displayed in Figure 39.



Figure 39 - Sequence flow line thickness and line color animation example

5.4.7. Resources animation

In order to animate resources performance metrics, the element fill color animation type should be used (Figure 40). A resource attribute that deserves to be animated is its state. The identified states that a resource can assume are “Idle”, ”Busy” and “Fail”. The first indicates that the resource is available to work; the second indicates that the resource is working and the last one indicates that the resource exists but is not presently available (e.g. employee is ill or machine requires repair). An icon was selected to present each resource state. The three icons are presented in Figure 41.



Figure 40 - Resource color animation type figure.



Figure 41 - Resource states icons (*From left to right: “Idle”, “Busy” and “Fail”*).

The combination of the resource presentation together with the display of its current state is represented in Figure 42.

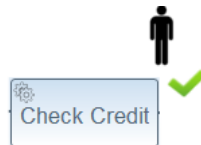


Figure 42 - Example of resource presentation with its status.

5.4.8. Queues animation

Queues should be animated with the element fill color animation type, as presented in Figure 43.

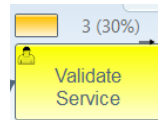


Figure 43 - Queue color animation example

However another graphical element can be manipulated to animate the queues: the queue size, or in other words the rectangle horizontal length. By increasing and decreasing the length to present the current state of the queue, it is possible to see if the queue is empty or already full. Besides these two animation types, the number of entities waiting on the queue should be presented in relative (when there is a maximum size defined) and absolute values, so the observer can have an immediate understanding of the queues state.

5.4.9. Process instances animation

The animation of process instances can be done by showing the movement of the instance from the activity of origin to the activity of destiny. Like aforementioned, a process instance can be presented with different icons depending on the desired semantics.

5.5. Animation views

In order not to overload the process model with animation elements and make the reading of the model harder the usage of animation views is proposed. The views purpose is to show only the information that the viewer wants to analyze. We propose the following three animation views:

- 1- Dynamic behavior view
- 2- Path load analysis view
- 3- Performance analysis view

5.5.1. Dynamic behavior view

The dynamic behavior animation view combines the element animations used to present the process dynamic behavior. This view incorporates the following animations:

- 1- Process instances animation

- 2- Resource animation (only shows the resource state icon)
- 3- Queue animation (shows the number of waiting instances and changes the size of the queue)

This animation view enables the animation for the three new graphical elements proposed. With this view the color animation for the queues and for the resources should not be shown. This view can be used to visualize the dynamic nature of a process. In other words, it can be useful to visualize the path each instance takes through the process model, the queues sizes and the resources states during the animation.

5.5.2. Path load analysis view

The path load analysis enables the animation for all the sequence flows and also the border line thickness and color animation for all the other animated graphical elements. This segment can be used, for example, to see the most traveled paths and also the most executed activities from all the workflow.

5.5.3. Performance analysis view

The performance analysis view enables the fill color animation for the activities, resources and queues. The fill color animation can be used to present any performance metric value of the animated element. This can be useful to visualize a performance metrics value for all elements of the same type and compare those elements performance. An example of a performance metric can be the average number of processed instances for a given resource. Using this view to visualize the aforementioned performance metric could be useful to understand which resource has the lower average number of processed instances.

5.6. BPMN 2.0 elements graphical rules

The rules presented in chapter 7.4 entitled “Use of Text, Color, Size, and Lines in a Diagram” of the BPMN 2.0 specification are analyzed in Table 43 to check if the animations proposed previously violate any of the rules.

Table 43 - Text, color, size and lines rules for BPMN

	BPMN 2.0 elements display Rules	Result
1.	BPMN elements (e.g. Flow objects) MAY have labels (e.g., its name and/or other attributes) placed inside the shape, or above or below the shape, in any direction or location, depending on the preference of the modeler or modeling tool vendor.	N/A
2.	The fills that are used for the graphical elements MAY be white or clear.	Respected
2.1	The notation MAY be extended to use other fill colors to suit the purpose of the modeler or tool (e.g., to highlight the value of an object attribute).	Respected
2.1.1	The markers for “throwing” Events MUST have a dark fill.	Respected
2.1.2	Participant Bands for Choreography Tasks and Sub-Choreographies that are not the initiator of the Activity MUST have a light fill.	N/A
3.	Flow objects and markers MAY be of any size that suits the purposes of the modeler or modeling tool.	N/A
4.	The lines that are used to draw the graphical elements MAY be black.	Respected
4.1.	The notation MAY be extended to use other line colors to suit the purpose of the modeler or tool (e.g., to highlight the value of an object attribute).	Respected
4.2	The notation MAY be extended to use other line styles to suit the purpose of the modeler or tool (e.g., to highlight the value of an object attribute) with the condition that the line style MUST NOT conflict with any current BPMN defined line style. Thus, the line styles of Sequence Flows, Message Flows, and Text Associations MUST NOT be modified or duplicated.	Respected

Table 43 shows that all rules presented in the specification of BPMN 2.0 have been respected in our animation proposal. In fact, some of the rules already indicate the possibility of extending the notation to use fill colors and other line styles, like mentioned in the rules 2.1, 4.1 and 4.2.

5.7. Summary

Concluding, in this chapter new graphical elements were presented and proposed to be used as animation elements in BPMN 2.0 models. These elements are Resource, Queue and Process Instance. In the case of Resource, the notion of resource role already existed in the BPMN 2.0 metamodel, so only a graphical element was proposed. In the case of the Queue element the concept was absent in the specification, so a metamodel extension is proposed as well as a graphical representation that can be used during model animation. The Process Instance notion already exists in the notation specification, but there is no corresponding metaclass in the BPMN 2.0 metamodel to represent it, so another metamodel extension as well as several graphical possibilities to show process instances during model animation, were proposed.

Animations for the BPMN elements and the new proposed elements were presented. Three types of animations were presented 1) border line thickness, 2) border line color, and 3) the color fill of the elements. These three animation types are already suggested in the BPMN 2.0 specification as it is possible to see in points 4.1, 4.2 and 2.1 displayed in Table 43.

Animation views were also proposed to serve as a filter on the presented animation, so the viewer sees only the animated elements he desires to analyze. Three views were presented: dynamic behavior view, path load analysis view and performance analysis view.

Chapter 6

Bpmn2Animator

Contents

6. Bpmn2Animator	72
6.1. Introduction	72
6.2. Bpmn2Animator use cases	72
6.3. Plug-in development.....	76
6.4. Bpmn2Animator package analysis	84
6.5. Summary	103

This chapter presents the prototype developed to show the BPMN animation capabilities proposed in the previous chapter. First, the functionalities are presented and second the prototype implementation is analyzed.

6. Bpmn2Animator

6.1. Introduction

Bpmn2Animator is our developed prototype that implements the BPMN extension proposed in the previous chapter. The technical purpose of Bpmn2Animator was to extend a BPMN2 modeling tool, and add animation features to it in order to present execution information. The extended BPMN2 modeling tool is the Bpmn2 Modeler, the one that was chosen through the application of our proposed OSS tool selection framework. The work related to the usage of the selection framework can be seen in chapter 4.

6.2. Bpmn2Animator use cases

Figure 44 presents the use cases for the Bpmn2Animator. Each one is described together with figures that show how the use case can be performed in the prototype.

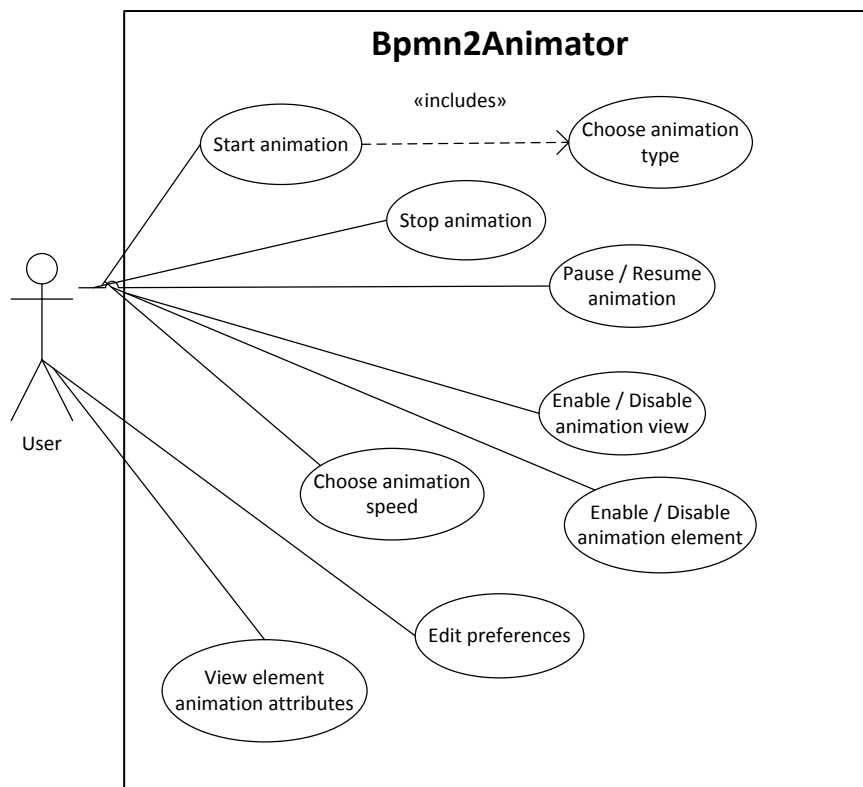


Figure 44 - Bpmn2Animator use case diagram

Start Animation

The user can start a new animation using the start button from the animation execution controls (Figure 45). To start a new animation it is necessary to choose its desired type among of the following alternatives:

Automatic event generator – starts an event generator that generates random events only for model demonstration;

Automatic event reader – starts an event generator that reads events from a CSV file;

Event reader with event manager – starts a window dialog where the user can choose a CSV file with events that will be read one by one, when required (Figure 47).



Figure 45- Animation execution controls

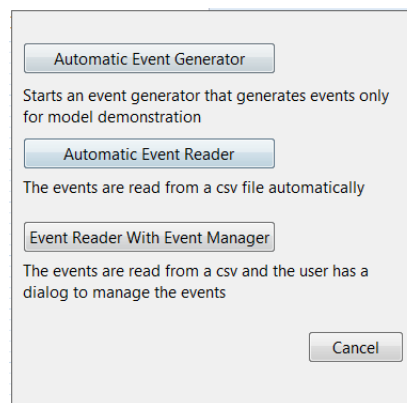


Figure 46 - Start animation dialog

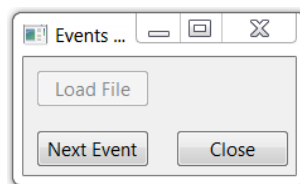


Figure 47 - Event Manager dialog

Pause/ Resume Animation

After starting an animation the user can pause it, by using the pause button from the animation execution controls, and resume it by using the start button when the user desires (see Figure 45).

Stop Animation

The user can stop an animation at any time by using the stop button from the animation execution controls. When the animation is stopped the model is refreshed appearing as it was before the start of the animation (see Figure 45).

Choose Animation Speed

The animation speed can be changed by the user in a 5 speed scale: -2; -1; 0; 1; 2. Smaller values correspond to lower speed animation and vice-versa. This value is not related in anyhow to the real time events, for instance, 2 does not mean that the animation will play 2 times faster than the real time present in the events logs.

Edit Preferences

It is possible to edit the preferences for the Bpmn2Animator. The user can choose to enable or disable the animation of the BPMN elements, as well as the animation for the additional elements proposed in the previous chapter. Disabling the animation of any element in the preferences will prevent the user from enabling the element during an animation execution. Besides the enablement of the elements animation, it is also possible to choose the instances animation mode which can be one of the following: “Move around” or “Appear only once”. The difference is that in the first mode the instance icon travels the sequence flows from one activity to the other whereas in the second mode the instance icon appears statically in the middle of the sequence flows for few seconds. Also in the preferences is possible to choose if the animation events should be logged or not, see Figure 48.

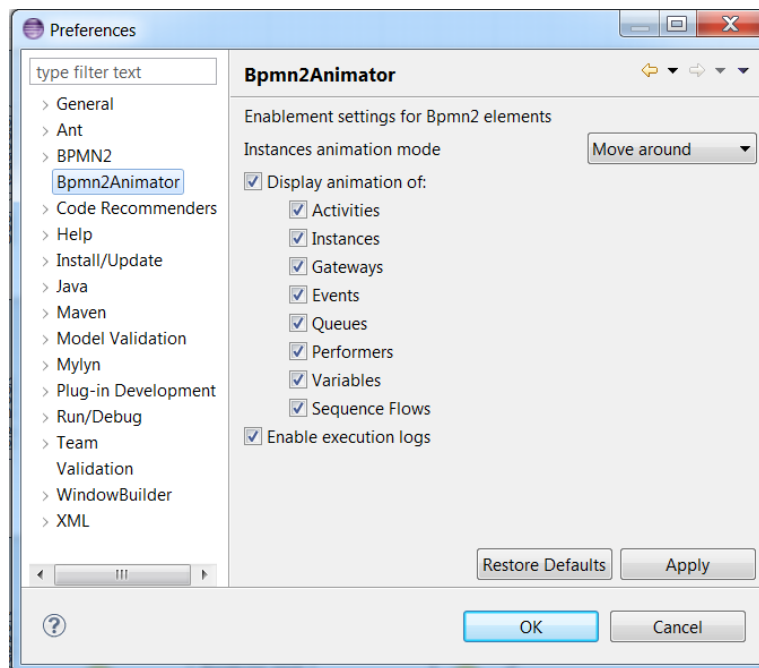


Figure 48 - Bpmn2Animator preferences window

Enable / Disable Element Animation

During an animation execution the user can select the elements he desires to see animated using the Bpmn2Animator animation management view, see Figure 49.

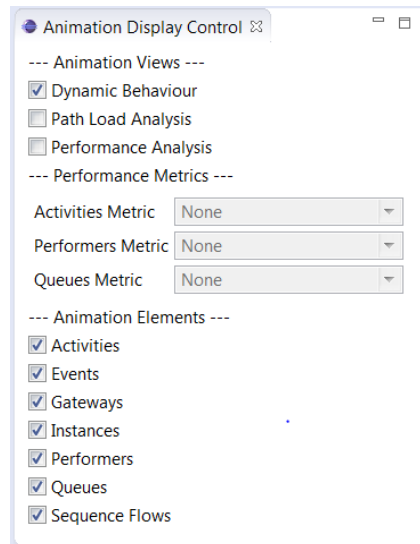


Figure 49 - Bpmn2Animator animation display control view

Enable / Disable animation segment

The animation views presented in the previous chapter were implemented in Bpmn2Animator. In order to control their presentation, the tool offers the “Animation Views” and “Performance Metrics” options in the Animation control view (see Figure 49). The “Performance Metrics” combo boxes are only enabled when the “Performance Analysis” animation view is enabled. Then these combo boxes allow the user to choose which metric is animated in the model elements.

View Elements Animation Attributes

The value of the metrics used to animate the model elements can be seen during an animation execution. In order to do that, the user just needs to select an element, navigate to the tab “Animation Attributes” located in the Properties view, and observe the existent fields.

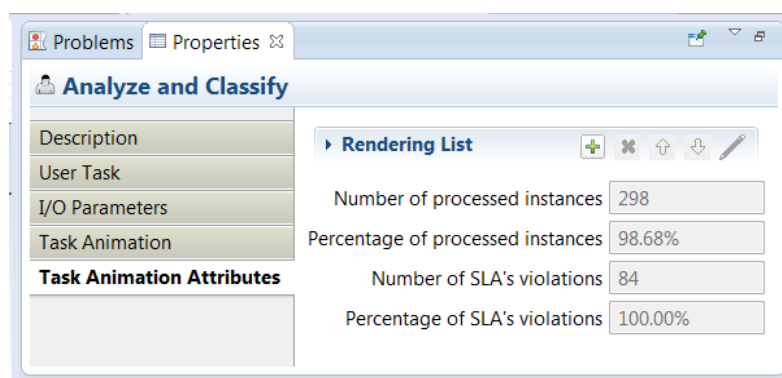


Figure 50 - Task animation attributes property tab

6.3. Plug-in development

Bpmn2Animator is a plug-in for Eclipse IDE that was developed to use a plug-in extension of Bpmn2 Modeler, an Eclipse plug-in modeling tool that supports the BPMN 2.0 notation. It depends on two other plug-ins, the first is the Eclipse Modeling Foundation, EMF, which is used to model the business layer and automatically generate the code. In Bpmn2 Modeler, the EMF was used to implement the notation specification. The second plug-in is Graphiti, a Graphical Tooling Infrastructure that takes care of the presentation layer. Basically, this framework provides an easy to use API for building graphical tools as Bpmn2 Modeler. The dependencies between the aforementioned plug-ins are presented in Figure 51.

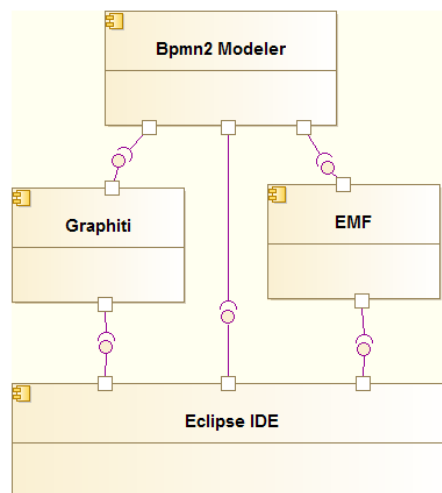


Figure 51 - Bpmn2 Modeler dependencies

The following objectives were established for Bpmn2Animator:

- Add new elements to the palette that could be used to show the human resources, the automated resources, queues and process instances;
- Add new attributes to the BPMN elements that could be used to store information about the execution, like the value of the execution metrics;
- Add user controls:
 - o Property tabs that present the animation attributes values;
 - o Start, pause, resume, stop and speed control in the menu toolbar;
 - o View that can be used to enable / disable the animation for each BPMN element type;
 - o Preferences window with options like the process instances animation type.

The elements that the Bpmn2Animator plug-in adds to the Eclipse IDE are represented in Figure 52 and Figure 53. The animation execution controls in the Menu toolbar, the “Animation Display

Control” view on the left, an example of a Property tab, are shown in Figure 52. The plug-in preferences page is presented in Figure 53.

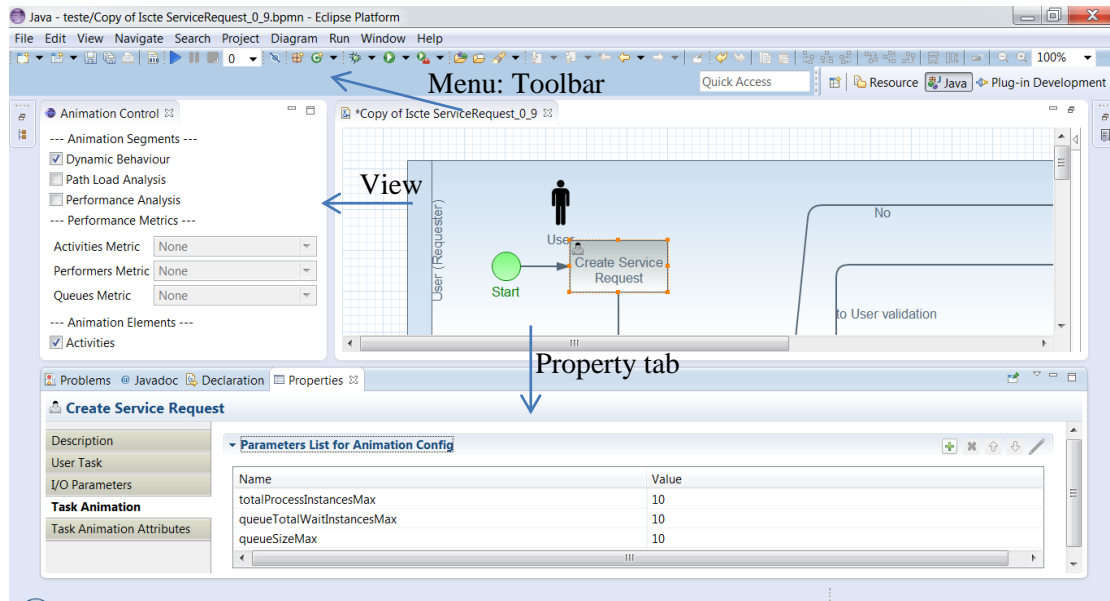


Figure 52 - Bpmn2Animator Eclipse contributions demonstration

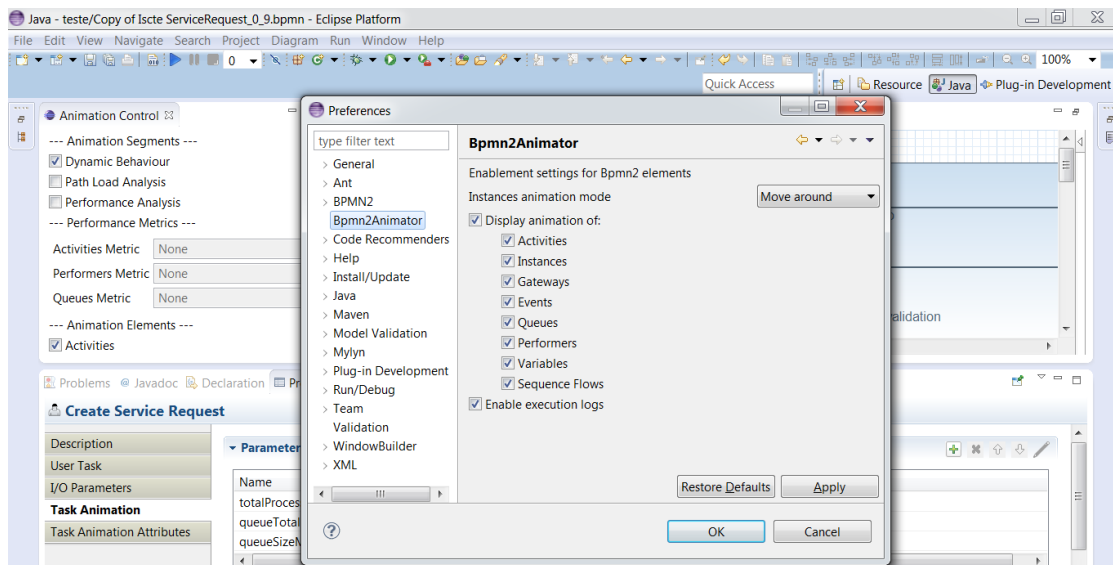


Figure 53 - Bpmn2Animator preferences page

6.3.1. Plug-in dependencies

Bpmn2Animator depends on other plug-ins to offer its functionalities. This means that in order to have Bpmn2Animator working in an Eclipse workspace we need to install the correct versions of the plug-

ins in which Bpmn2Animator depends. Following we present the needed plug-ins and the links where they can be obtained from.

Table 44 - Bpmn2 Modeler plug-ins

Plug-in	Version
org.eclipse.bpmn2.editor	0.7.0
org.eclipse.bpmn2.modeler.ui	0.2.7
org.eclipse.bpmn2.modeler.core	0.2.7

The plug-ins presented in Bpmn2Animator depends on other plug-ins to offer its functionalities. This means that in order to have Bpmn2Animator working in an Eclipse workspace we need to install the correct versions of the plug-ins in which Bpmn2Animator depends. Following we present the needed plug-ins and the links where they can be obtained from.

Table 44 plug-ins are installed with the Bpmn2 Modeler that can be found in: <http://download.eclipse.org/bpmn2-modeler/updates/juno/0.2.7/>

Table 45 - Graphiti plug-ins

Plug-in	Version
org.eclipse.graphiti	0.9.1
org.eclipse.graphiti.ui	0.9.1

The plug-ins presented in Table 45 are part of the Graphiti plug-in and can be installed from: <http://download.eclipse.org/graphiti/updates/0.9.2/>

Table 46 - EMF plug-ins

Plug-in	Version
org.eclipse.emf.transaction	1.4.0

The EMF plug-in presented in Table 46 is installed when EMF is installed from: <http://download.eclipse.org/modeling/emf/updates/>

Table 47 - Eclipse IDE plug-ins

Plug-in	Version
org.eclipse.e4.ui.di	0.10.1

org.eclipse.core.resources	3.8.1
org.eclipse.ui.ide	3.8.2
org.eclipse.ui.views.properties.tabbed	3.5.300

The plug-ins presented in Table 47 do not have to be downloaded because they already come with Eclipse IDE Juno in the “Eclipse for RCP and RAP Developers” package solution, which was the version used to develop the Bpmn2Animator.

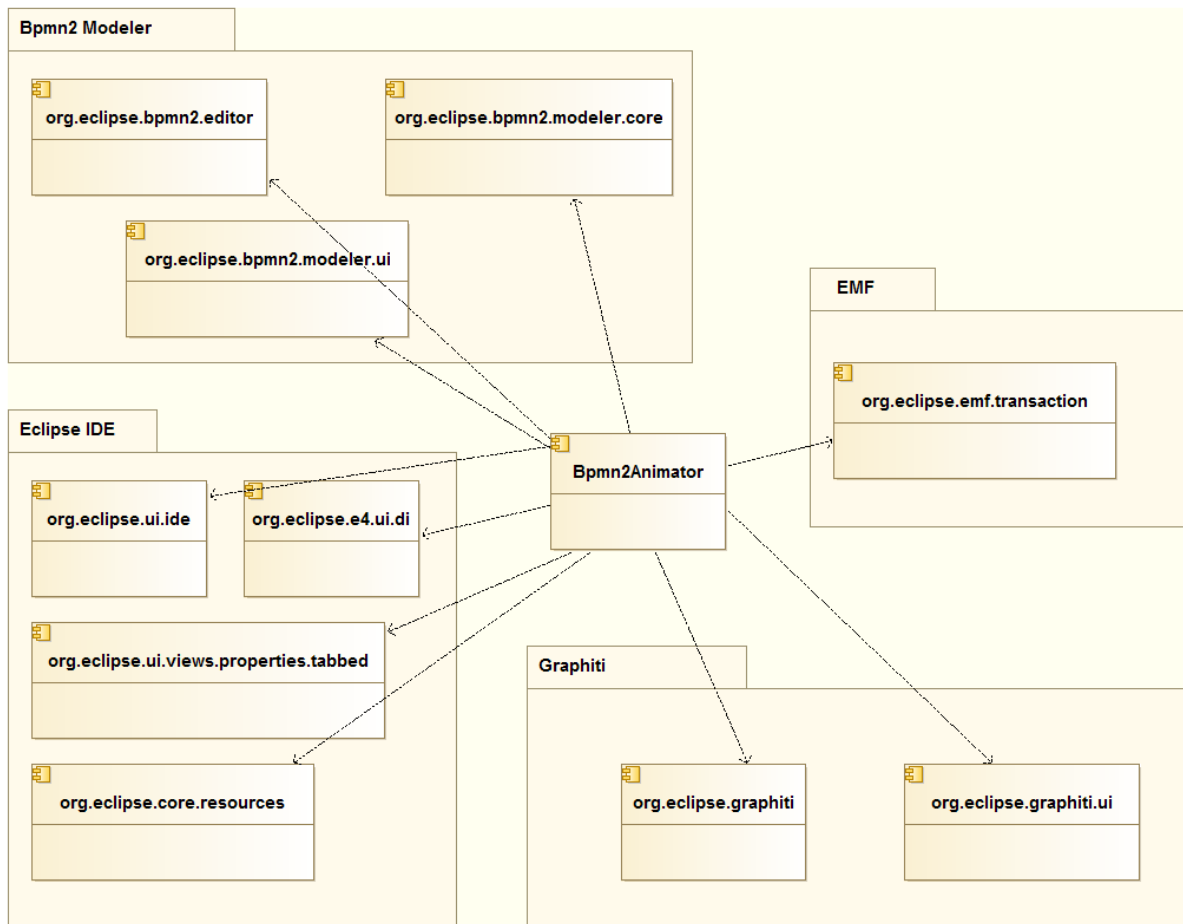


Figure 54 - Bpmn2Animator plug-ins dependencies

6.3.2. Plug-in Extensions

Eclipse provides the concept of plug-in extension that enables new plug-ins to contribute functionality to already existing plug-ins. This can be done through the usage of plug-in extensions points that are made available by the existing plug-in and used by the new plug-in. Bpmn2Animator uses several extension points from the plug-ins that were presented in the Plug-in dependencies section. The used extension points can be observed in Figure 55.

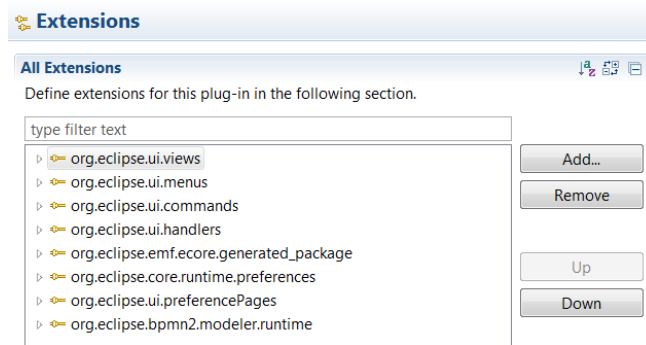


Figure 55 - Extension points used by Bpmn2Animator

Table 48 presents the extension points that were used in order to develop Bpmn2Animator contributions to the IDE.

Table 48 - Bpmn2Animator used Extension points and their description

Extension Point	Description
org.eclipse.ui.views	Extension point used to contribute with the “Animation Display Control” (Figure 49).
org.eclipse.ui.menus	Extension point used to present the animation execution controls in the menu toolbar (Figure 45).
org.eclipse.ui.commands	Extension point that defines the commands that are executed when the animation execution controls are used. The animation speed control is the only one that does not use this extension point.
org.eclipse.ui.handlers	This extension point is used to define the handlers used by the commands defined in the “org.eclipse.ui.commands” extension point.
org.eclipse.emf.ecore.generated_package	Extension point used to define the <i>ModelPackage</i> and the model code generator that is used by the plug-in.
org.eclipse.core.runtime.preferences	Extension point used to define the class used to initialize the preferences values. This class is the one where the preferences default values are defined.
org.eclipse.ui.preferencePages	Extension point used to define the preference page that appears in the Windows -> Preferences menu.

org.eclipse.bpmn2.modeler.runtime

This extension point is the essential part of the plug-in because it is in the Bpmn2 Modeler runtime extension, where the new functionalities are added to the model.

6.3.3. Bpmn2 Modeler runtime extension

Several extension points were used to create Bpmn2Animator, but the most important is the “org.eclipse.bpmn2.modeler.runtime” extension point. This extension point allows other plug-ins to contribute with new functionalities to the Bpmn2 Modeler. For example, a plug-in can contribute with new activities or new attributes to already existent Bpmn2 plug-in elements. Basically, the plug-in using this extension point creates a new runtime that can be used later when using the Bpmn2 Modeler, being only necessary to define the target runtime property in the Bpmn2 Modeler preferences window.

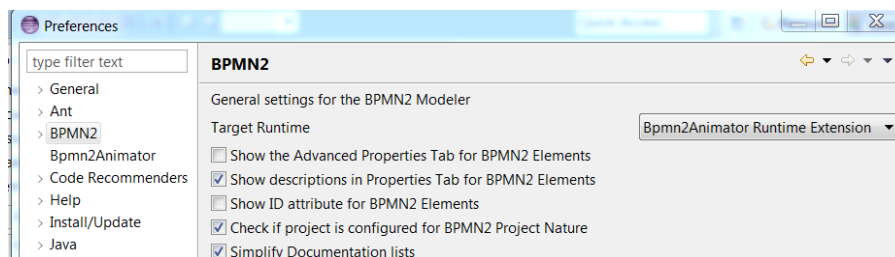


Figure 56 - Bpmn2 Modeler preferences window with Target Runtime

Bpmn2Animator uses several extension elements from this extension point in order to contribute with new elements and models extension to the Bpmn2 Modeler. These extension elements are presented in Figure 57 and each extension element is described in Table 49.

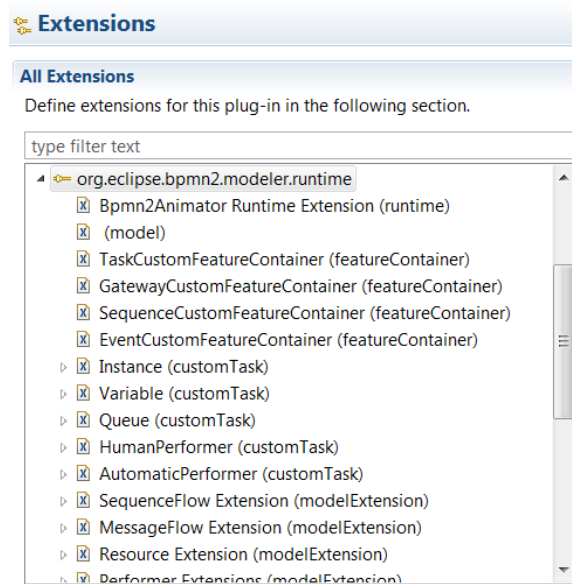


Figure 57 - Extension elements from Bpmn2 Modeler used by Bpmn2Animator.

Table 49 - org.eclipse.bpmn2.modeler.runtime extension elements description.

Type	Description
runtime	Extension element used to define the runtime extension details.
model	Extension element used to define the information about the model that will be used in the runtime extension.
<i>featureContainer</i>	A <i>featureContainer</i> is responsible for containing the graphical element behavior information. In other words, is in the feature container that is defined what should happen to the graphical element and linked model elements when a graphical model is added, updated or removed from the model. In the plug-in, the <i>featureContainer</i> holds an instance of <i>AnimationCustomFeatures</i> that is used to control the graphical elements animation.
<i>customTask</i>	The <i>customTask</i> extension element is used to define a new custom task that will be available in the modeler when using the runtime extension. It is used to define the <i>featureContainer</i> , the element type that it extends and a <i>propertyTab</i> that will be used by the <i>customTask</i> .
<i>modelExtension</i>	This extension element is used to add new attributes (defined in the extension model) to a Bpmn2 plug-in element. These new attributes will be added to the <i>extensionValues</i> properties of the Bpmn2 plug-in elements when using the runtime extension.

<i>propertyTab</i>	Extension element used to define a new <i>propertyTab</i> that will be presented in the graphical element properties view. In Bpmn2Animator plug-in it is used to add new <i>propertyTabs</i> that are used to present the animation attributes and animation configuration parameters of the graphical elements.
<i>modelEnablement</i>	Extension element that defines the graphical elements for a given model type (process, collaboration or conversation) that are presented in the modeler palette when using the runtime extension.

One good example of usage of these extension elements is the *HumanPerformer* which is a *customTask* extension element. The details of the extension element are presented in Figure 58.

```
<customTask
  category="Animator"
  description="HumanPerformer"
  featureContainer="org.quasar.bpmn2.animator.featureContainers.PerformerFeatureContainer"
  id="org.quasar.bpmn2.animator.humanPerformer"
  name="HumanPerformer"
  propertyTabs="org.quasar.bpmn2.animator.performerTab"
  runtimeId="org.quasar.bpmn2.animator.runtime"
  type="TextAnnotation">
</customTask>
```

Figure 58 - *HumanPerformer CustomTask* extension details

By using this extension element with these details it is possible to show a new element in the modeler palette in the new category “Animator” with the description of “*HumanPerformer*”. The type attribute defines that this *CustomTask* will be of the type *TextAnnotation*, so it will inherit all *TextAnnotation* properties. The *featureContainer* property defines the class that is used as *featureContainer* for this new *CustomTask*. In the end we get the new defined *HumanPerformer* element available in the Bpmn2 Modeler palette as it is possible to see in Figure 59.

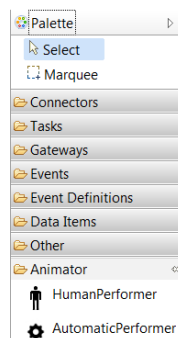


Figure 59 - Bpmn2 Modeler palette with the “Animator” category

When we add a new *HumanPerformer* element to a task, the *featureContainer* class overrides the *decorateShape* method which is called by the *AddTextAnnotationFeature* class and paints the new *HumanPerformer* in the intended way. The final result is presented in Figure 60.

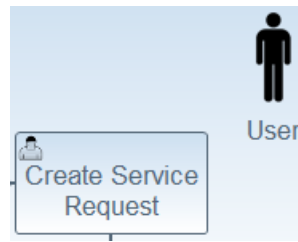


Figure 60 - *HumanPerformer* usage example.

The *HumanPerformer* is just one example of several extension elements used in *Bpmn2Animator*. All of these extensions are described in Table 61.

6.4. Bpmn2Animator package analysis

Table 50 - *Bpmn2Animator* package descriptions

Package Name	Description
org.quasar.bpmn2.animator	Contains the activator for the plug-in. It contains also the <i>ImageProvider</i> class that manages all images used in the <i>Bpmn2Modeler</i> runtime extension.
org.quasar.bpmn2.animator.animationFeatures	Contains the animation features that manipulate the model graphical elements.
org.quasar.bpmn2.animator.Bpmn2AnimatorModel	Package generated automatically by EMF. It contains model related classes.
org.quasar.bpmn2.animator.Bpmn2AnimatorModel.impl	Package generated automatically by EMF. It contains model related classes.
org.quasar.bpmn2.animator.Bpmn2AnimatorModel.util	Package generated automatically by EMF. It contains model related classes.
org.quasar.bpmn2.animator.contexts	Contains custom contexts that were created to pass more information to the <i>featureContainer</i> and also the <i>AnimationCustomContext</i> that is sent to <i>animationFeatures</i> .
org.quasar.bpmn2.animator.diagramAnimation	Contains the classes responsible for managing the graphical elements presentation and animation. Also contains the <i>DiagramAnimationFeatureExecuter</i> that manages the execution of the <i>animationFeatures</i> .

org.quasar.bpmn2 animator.dialogs	Contains classes that define dialogs used in the plug-in like for example the animation type dialog.
org.quasar.bpmn2 animator.engines	Contains the engines classes that are used to animate the model.
org.quasar.bpmn2 animator.eventGenerators	Contains the classes that support the generation of events that are sent to the animation <i>Engine</i> .
org.quasar.bpmn2 animator.featureContainers	Contains the classes that implement the <i>FeatureContainers</i> for the new created elements like the <i>Queue</i> or <i>Instance</i> .
org.quasar.bpmn2 animator.handlers	Contains the classes that implement the handlers for the animation controls.
org.quasar.bpmn2 animator.modelMetrics	Contains the classes responsible for managing the model animation attributes.
org.quasar.bpmn2 animator.preferences	Contains the classes responsible for the preferences window and its functionalities.
org.quasar.bpmn2 animator.propertyTabs	Contains the classes that implement the <i>propertyTabs</i> used to show the extension attributes added to the model elements.
org.quasar.bpmn2 animator.utils	Contains utilities classes that are used frequently by other classes. It contains a class for pictogram related utilities and other for a model related utilities.
org.quasar.bpmn2 animator.views	Contains the classes responsible for the animation display control view.

In the following sections the Bpmn2Animator plug-in implementation is explored and described with the objective of explaining the internals of the solution. To accomplish this task many diagrams were created with the help of modelGoon¹⁵, a plug-in for Eclipse IDE that enables the creation of diagrams from existing source code. The diagrams created were UML class diagrams, package diagrams, interaction diagrams and sequence diagrams. In Figure 61 is showed a package diagram that contains all Bpmn2Animator packages. The diagram shows the packages as rectangles with each package name and the existing packages dependencies as blue arrows and red arrows when there is a cyclic dependency. Basically a package depends on other package when classes of the first package use classes of the second package.

¹⁵ <http://www.modelgoon.org>

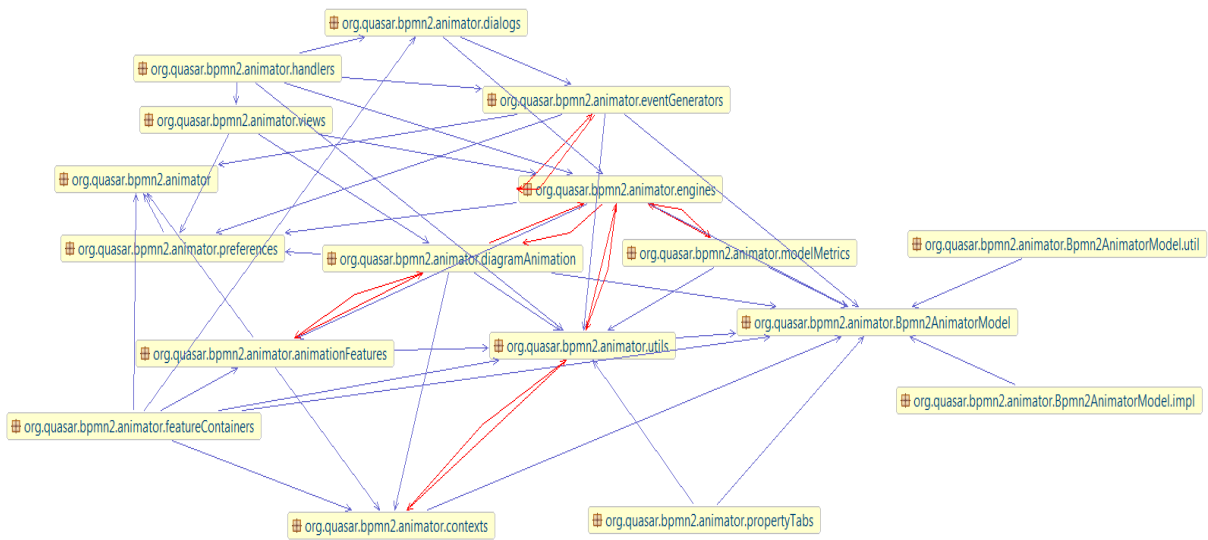


Figure 61 - Bpmn2Animator package diagram (all packages)

Figure 61 presents the dependencies among all Bpmn2Animator packages. To improve readability, the previous package diagram will be split into three parts: animation engines, model and user controls. Each package is also briefly described in

Table 50.

6.4.1. Bpmn2Animator animation engines

In this section the animation engines are explored in depth to explain how the engines work and which main objectives are for each one.

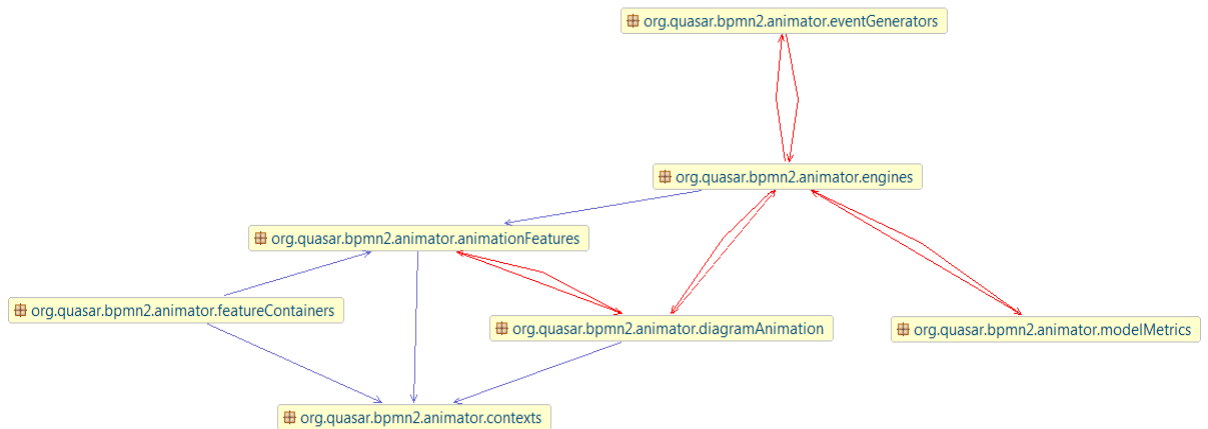


Figure 62 - Package dependencies diagram centered on the engines package.

The package dependencies diagram presented in Figure 62 show that the engines package has dependencies on the *diagramAnimation*, on the *modelMetrics*, on the *eventGenerators* and on the

animationFeatures. It is a central package because it contains the *AnimationEngine*, the class that is supposed to be used as the entrance point for animation instructions. The classes in the engines package communicate with classes of the *modelMetrics* package to request an update to the model metrics and also with the *diagramAnimation* in order to send information to be displayed in the graphical elements of the model. The event generator is started, paused and stopped by instructions sent from the *AnimationEngine* class contained in the engines package, which explains the existent dependency of engines package on the *eventGenerators*. Finally, the engines package has dependencies with the *animationFeatures* package because the *DiagramAnimationEngine* class contained in the engines package initializes the objects from the *diagramAnimation* package, that have as argument, objects from classes contained in the *animationFeatures* package. The *eventGenerators* classes generate all the events information that is then sent to the engines. The engines classes, on the other hand, receive the events and ask the *modelMetrics* and *diagramAnimation* to process them and finally animate the model. The packages will be now described in depth with class diagrams.

Engines Package

Figure 63 presents a class diagram of the engines package. The *AnimationEngine* class is the entrance point of events to be displayed in the animated model. The *ModelAnimationEngine* class is responsible for managing the model objects animation information, whereas the *DiagramAnimationEngine* is responsible for preparing the animations for the model graphical elements. The *StopCoordinator* is used as a barrier for the pause and stop controls, so when one of these two controls is pressed all threads registered in the *StopCoordinator* wait until the animation is resumed, or in the case of the stop command, wait until all registered threads have finished doing what they were doing.

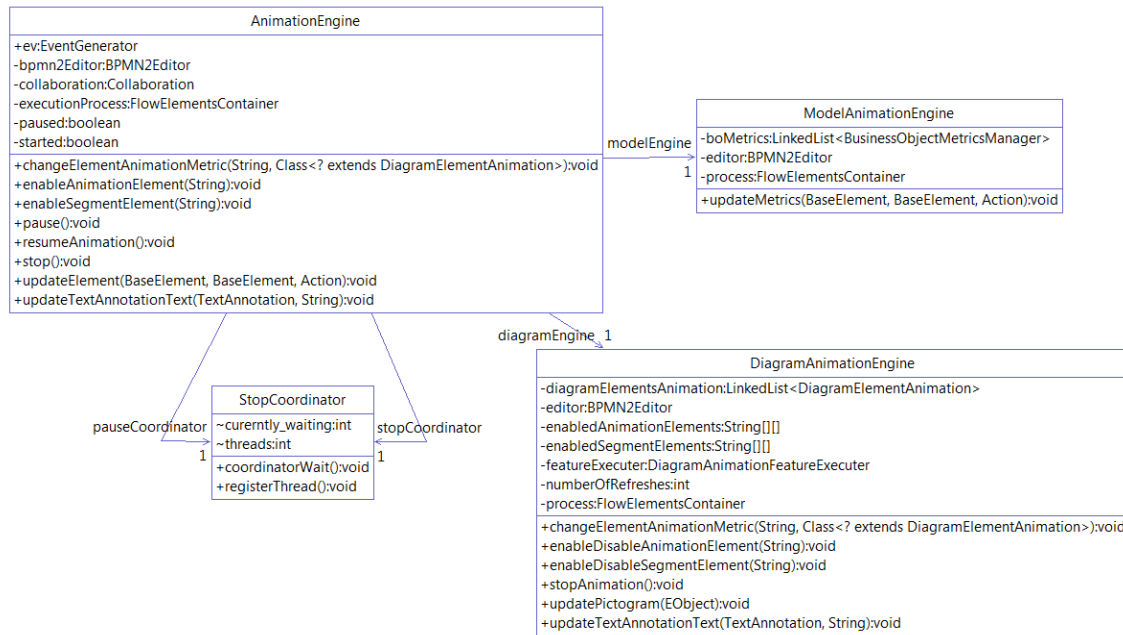


Figure 63 - Engines package class diagram.

AnimationEngine class

The idea of event receiver and animation controller that the *AnimationEngine* has can be perceived by observing Figure 64, where the methods invocations are all made by the *AnimationEngine* are represented. We followed the MVC (Model-view-controller) architecture where the *ModelAnimationEngine* class represents the model, the *DiagramAnimationEngine* class represents the view and the *AnimationEngine* class represents the controller.

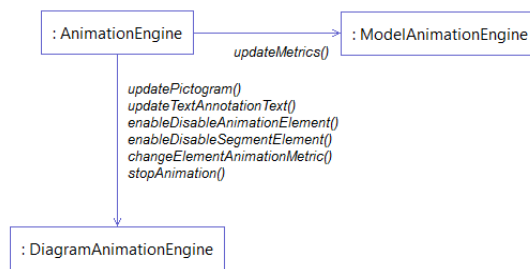


Figure 64 - AnimationEngine interaction diagram

The Singleton design pattern was implemented for the *AnimationEngine* to make an instance of the *AnimationEngine* available to be used easily. With this implementation an instance of the animation engine can be easily obtained like showed in the following piece of code:

```
AnimationEngine ae = AnimationEngine.getInstance();
```

After being able to successfully obtain an instance of the *AnimationEngine*, it is possible to start asking for updates to the model elements by using the method *updateElement(BaseElement baseElement, Action action)* like in the following example:

```
animationEngine.updateElement(queue, Action.QUEUEWAITIN);
```

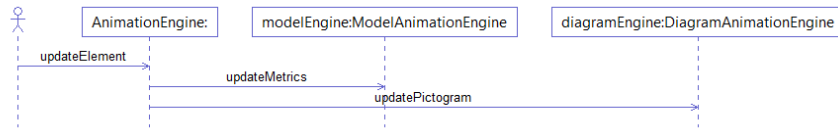


Figure 65 - Sequence diagram of the *AnimationEngine* *updateElement* method.

Figure 65 shows an UML sequence diagram of the methods invocations triggered by the call to the class.

- *DiagramAnimationEngine*

As aforementioned, the *DiagramAnimationEngine* has the responsibility of animating the graphical elements in the model. In order to do that, it interacts with classes from the *diagramAnimation* package as shown in Figure 62. The class diagram for the *diagramAnimation* package is presented in the Figure 66.

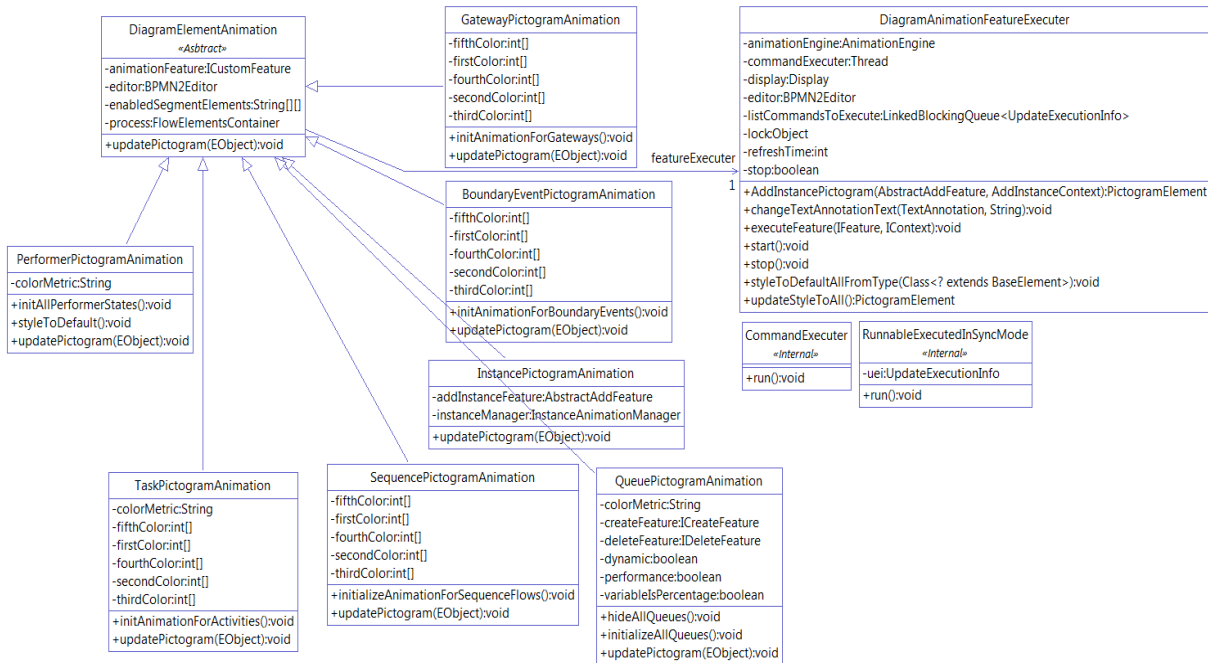


Figure 66 - *diagramAnimation* package class diagram.

In the *diagramAnimation* package there are two essential classes that must be explained. The first one is the *DiagramElementAnimation* abstract class. The concept behind this class is that there should be

one class responsible for the graphical animation of each BPMN2 element type, like activity or gateway. Following that concept, the classes that extend the *DiagramElementAnimation* were created. In Figure 66 it is possible to see the classes that extend it. It is possible to see that the only method that the abstract class defines is the *updatePictogram* method, although the classes that extend it have more methods defined. This happened because there are some actions that are required to initialize the elements of a certain type, for example the queues only appear in the diagram in the start of the animation, so in order to create the queues and show them in the diagram the *initializeAllQueues* method is used by the *DiagramAnimationEngine*. The second essential class is the *DiagramAnimationFeatureExecuter*. This class executes the features that produce visual changes on the model diagram. As it is possible to see in Figure 67 the *executeFeature* method is invoked by the classes that extend *DiagramElementAnimation*, which results in the insertion of the feature to execute in a queue. Then, the *CommandExecuter* thread contained in the *DiagramAnimationFeatureExecuter* will ask the Bpmn2 Modeler editor to execute the feature. The mechanism implemented with the *DiagramAnimationFeatureExecuter* was created to prevent concurrency problems in the model updates. This implementation ensures that only one feature is executed at a time, because we have a central point that asks for the execution and makes a small interval between each one.



Figure 67 - Sequence diagram for *updatePictogram*.

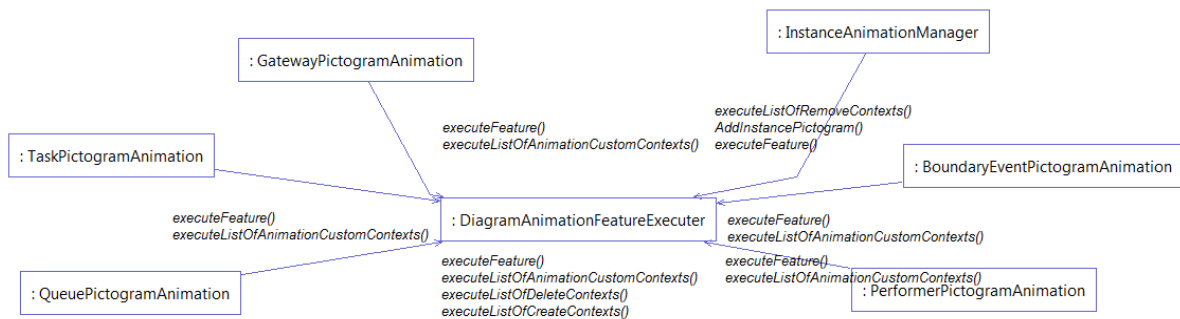


Figure 68 - *DiagramAnimationFeatureExecuter* interaction diagram

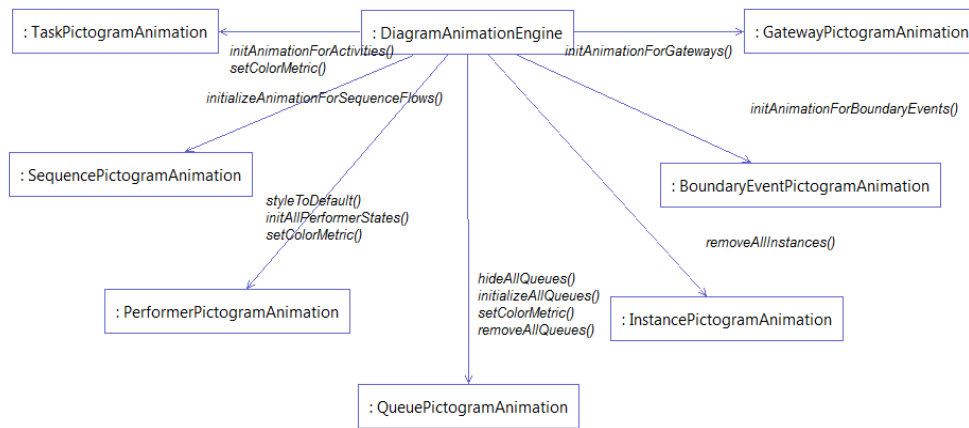


Figure 69 - *DiagramAnimationEngine* interaction diagram

The *DiagramAnimationEngine* interacts with the classes that extend the *DiagramElementAnimation*, with the exception of the *InstancePictogramAnimation*, at the start of the animation, to initialize the animation for the diagram graphical elements. The *DiagramAnimation* does this by invoking the methods which name begins by “initialize” or “init”. In the case of the queues the *QueuePictogramAnimation* will actually create a new model and graphical elements to show the queues in the diagram. In the case of tasks, the *TaskPictogramAnimation* will only change certain aspects of the corresponding graphical element.

Process instances diagram animation

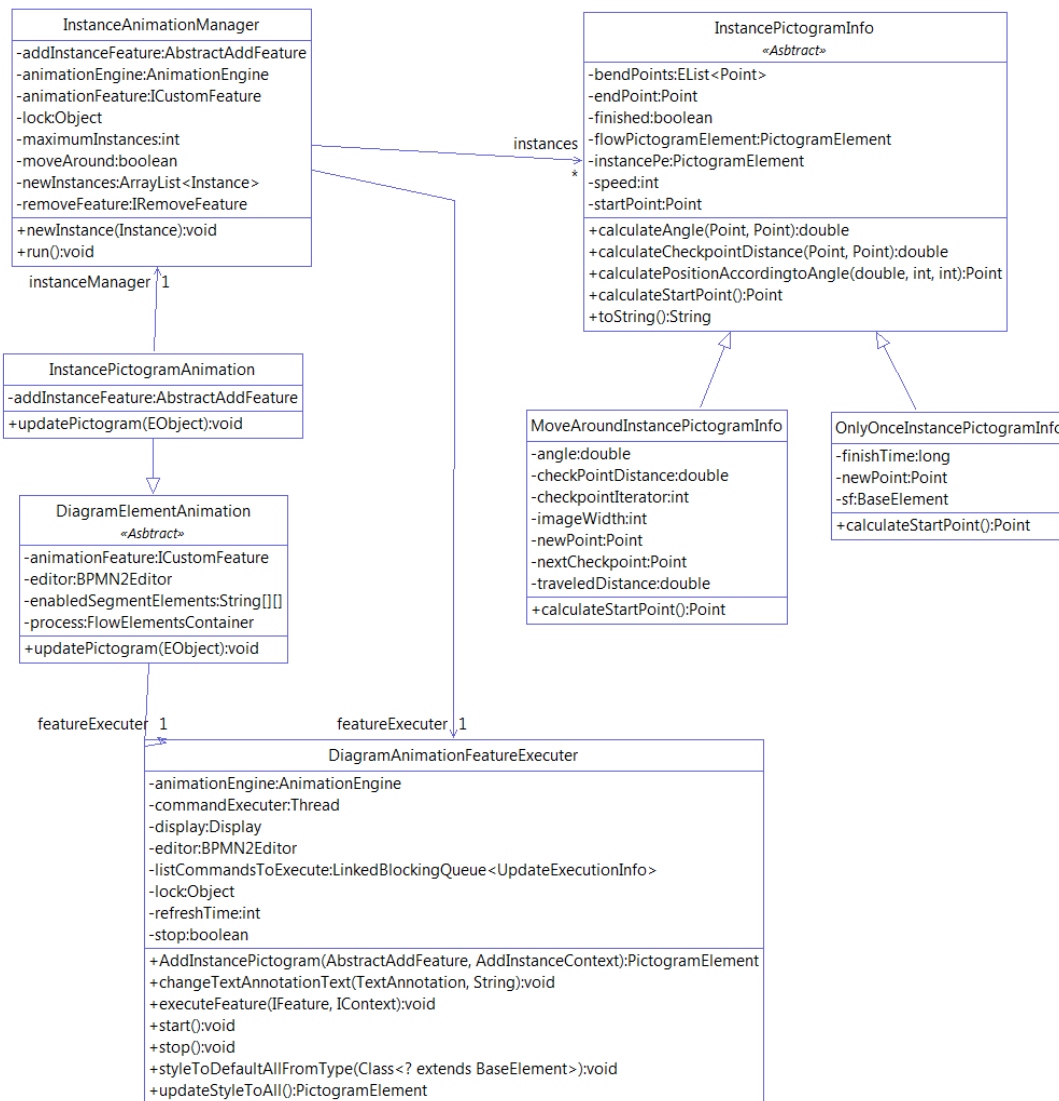


Figure 70 - Process instance animation class diagram

Although slightly more complex, the mechanism used for process instances animation is based on the same mechanism that is used to animate all the other elements. This means that there is an *InstancePictogramAnimation* that extends the *DiagramElementAnimation* as it is possible to see in Figure 70. The difference is in the *InstanceAnimationManager*, the class responsible for managing the process instances animation. The latter is responsible for adding new instances to the diagram, for making them travel through the sequence flows and for removing the instances from the diagram when they reach the end of the sequence flow. For each process instance is created an instance of *MoveAroundInstancePictogramInfo* or *OnlyOnceInstancePictogramInfo* which are both extending classes of *InstancePictogramInfo*. The function of these classes is to store the information of a running instance like the *startPoint* of the sequence flow, its *bendPoints* (also seen as checkpoints) and the end

point that represent the end of the sequence flow. The difference between the classes *MoveAroundInstancePictogramInfo* and *OnlyOnceInstancePictogramInfo* is that the first is used for the “Move around” option and the second one is used for the “Appear only once” option of the “Instances animation mode” that exist in the *Bpmn2Animator* preferences window (see Figure 48).

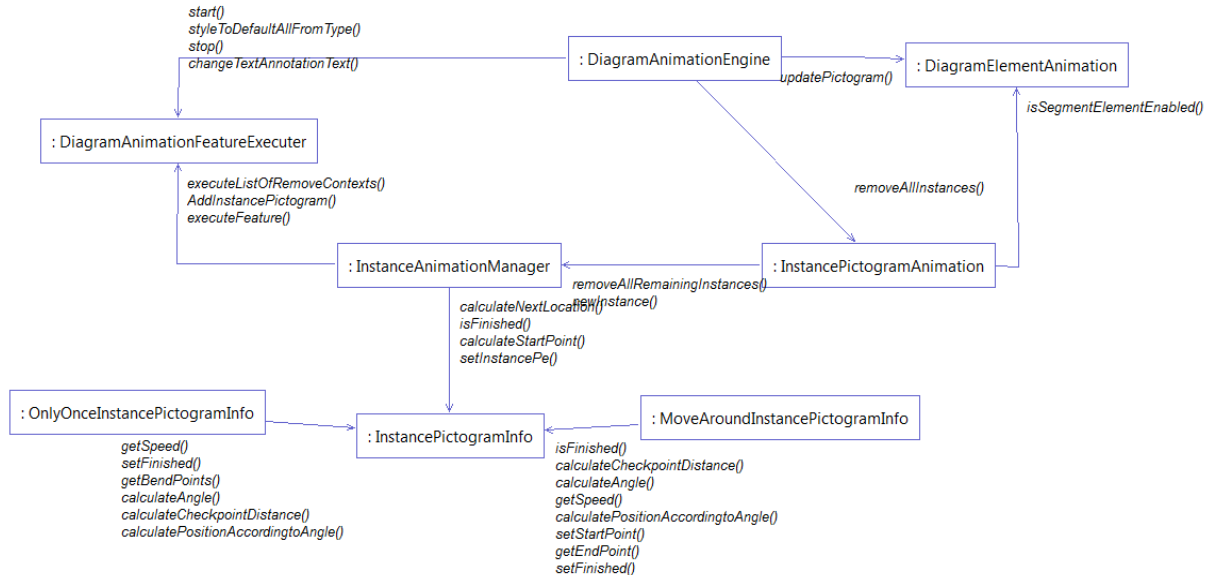


Figure 71 - Process instance animation interaction diagram

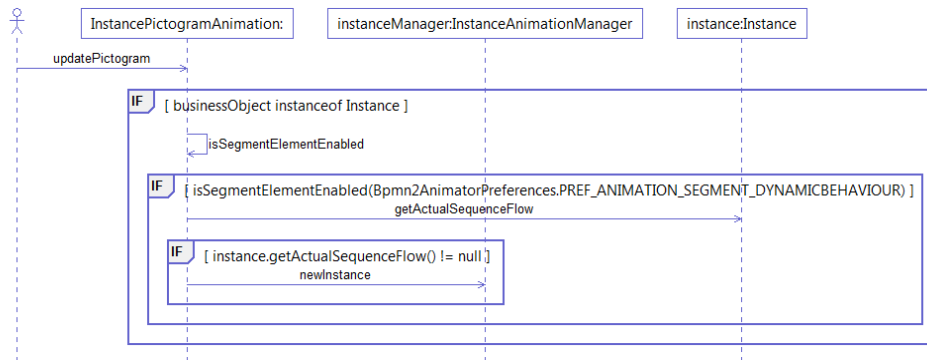


Figure 72 - Instance animation sequence diagram

A request for the animation of a process instance has its origin in the event generator that invokes the *updateElement* method of the *AnimationEngine*. It then asks the *DiagramAnimationEngine* to resolve the request and the *updatePictogram* method, from the *InstancePictogramAnimation*, is invoked. Here is where the process instance mechanism starts working. As it is possible to observe in Figure 72 the *updatePictogram* does some validations and then invokes the *newInstance* method from the *InstanceAnimationManager* as it is also possible to see in the interaction diagram presented in the Figure 71. The new instance is then added to the queue until the thread of the

InstanceAnimationManager gets the request and starts the animation for it, by creating a new instance of *InstancePictogramInfo* and immediately asking to create a new pictogram in the model. Afterwards, the *InstanceAnimationManager* will ask the *InstancePictogramInfo* instance to calculate its next position, will create a new feature with the pictogram update information and will ask the *DiagramFeatureExecuter* to execute the feature. This will happen until the instance reaches the end point and is removed from the model.

- *ModelAnimationEngine*

The *ModelAnimationEngine* is similar to the *DiagramAnimationEngine*, but instead of managing the presentation of the graphical elements in the model, the former engine manages the model attributes that are used later by the *DiagramAnimationEngine*. The engine adds attributes to each BPMN element in the model at the start of the animation, which it maintains during the animation execution by calculating the necessary attributes for each event.

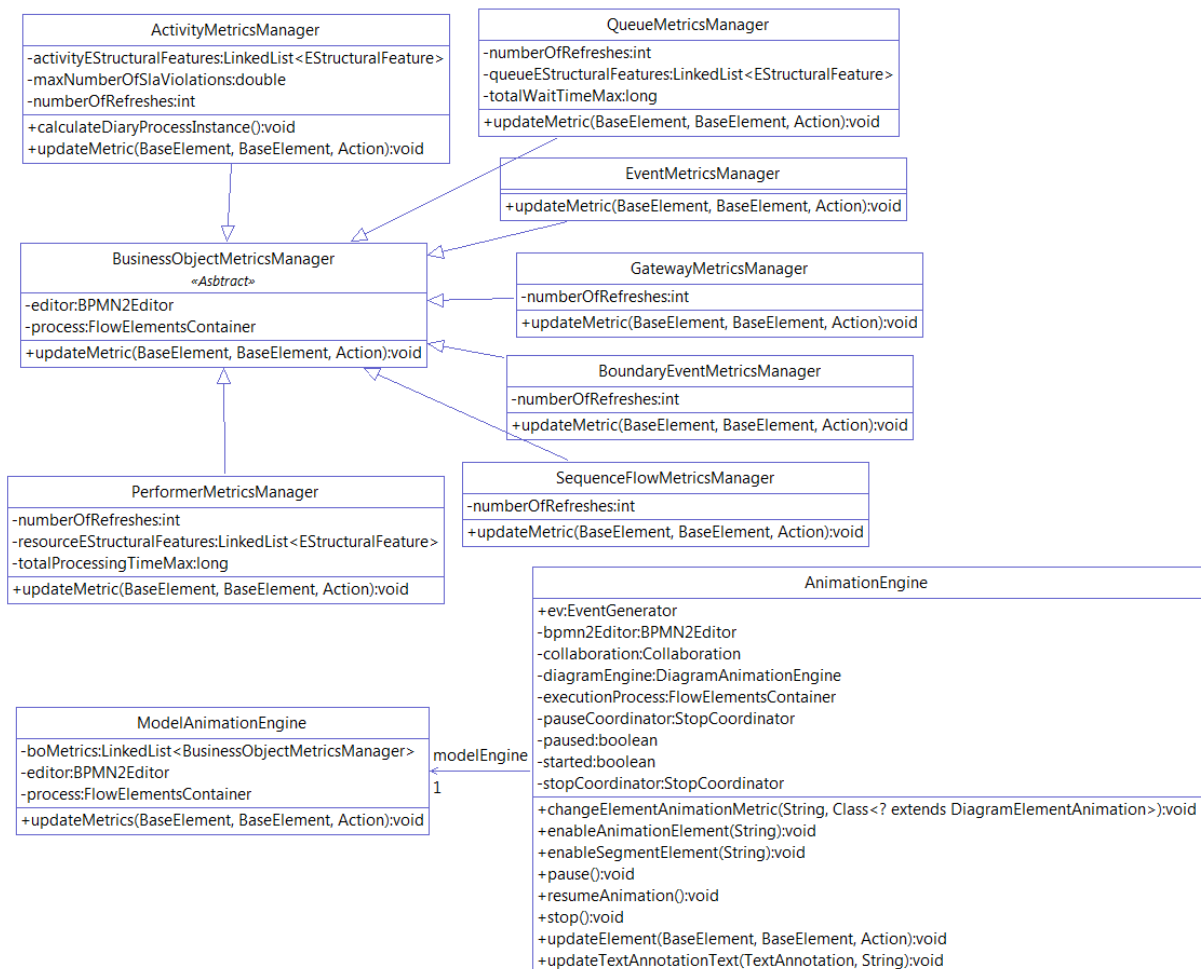


Figure 73 - *ModelMetrics* package class diagram

The class diagram presented in Figure 73 is very similar to the one presented in Figure 66. There is an abstract class called *BusinessObjectMetricsManager* that is extended by other classes, each one responsible for a specific type of BPMN element. For example, the *ActivityMetricsManager* is responsible for managing the attributes of all activities.

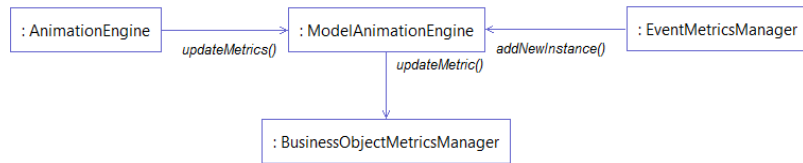


Figure 74 - *ModelAnimationEngine* interaction diagram

The interactions that exist for the *ModelAnimationEngine* are few and easily explained. The engine receives events as arguments to the *updateMetrics* method that is invoked by the *AnimationEngine*. The *ModelAnimationEngine* figures what *BusinessObjectMetricsManager* instance should be used to process the received event and then asks it to update the metrics using the *updateMetric* method of the *BusinessObjectMetricsManager*. The interaction with the *EventMetricsManager* exists because the total number of created instances is stored in an attribute of the engine called *numberOfInstances* that is updated by request of the *EventMetricsManager* when it is processing a start event.

6.4.2. Bpmn2Animator model

The original model that is used by the Bpmn2 Modeler plug-in does not contain classes that the *Bpmn2Animator* needs, like for example the class that defines the instance or the queue. So, in order to add these classes and other attributes that exist in the Bpmn2Animator plug-in it was necessary to create a model that defines the elements and extends the original BPMN2 model. To accomplish that task the EMF framework was used (Steinberg et al. 2009). Basically, this framework allows the developer to define a model in an Ecore extension file, by using a special editor as presented in Figure 75, create a genmodel file, which contains additional information for the code generation, based on the defined model, and finally use the last file to generate the model source code (Steinberg, Budinsky, Paternostro and Merks 2009).

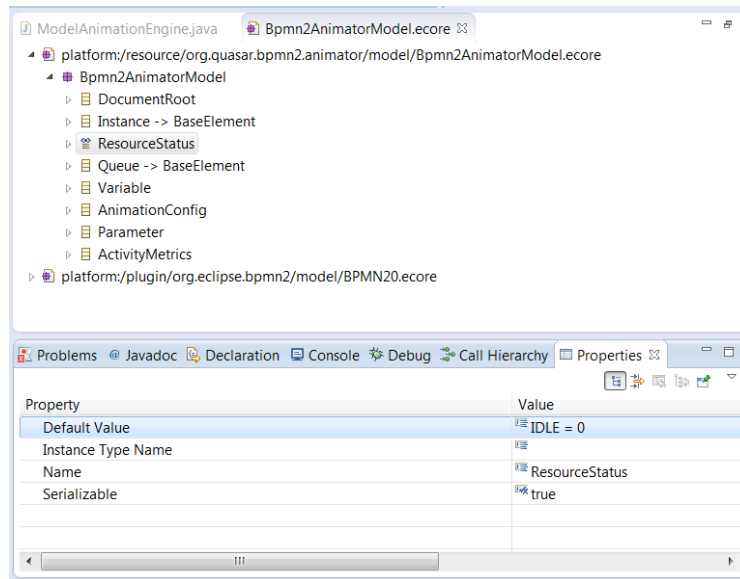


Figure 75 - EMF model editor

In the `Bpmn2Animator` the model code was generated to the `Bpmn2Animator` package. The classes of this package are used in almost every package, it makes almost all other as it is possible to observe in Figure 76.

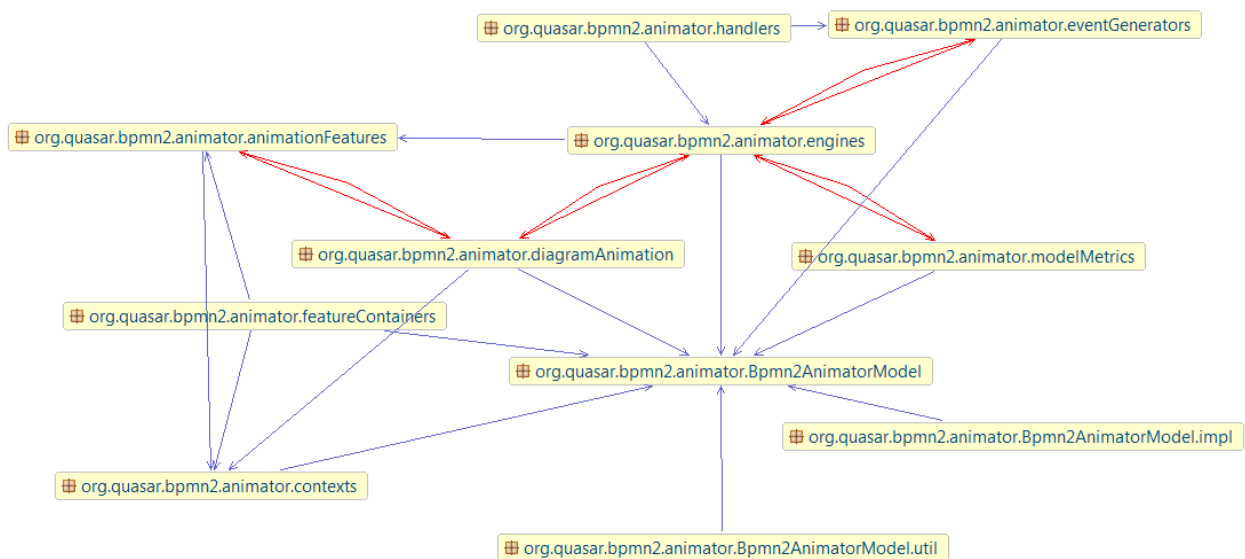


Figure 76 - `Bpmn2AnimatorModel` package diagram

6.4.3. Bpmn2Animator user controls

`Bpmn2Animator` offers several user controls that can be used to have control over what the animation is presenting. Recalling what was presented in section 6.3 and in this one, the controls that `Bpmn2Animator` offers are the following:

- Animation execution controls (start, pause, resume, stop and animation speed);
- Start animation dialog;
- Preferences window;
- Animation control view;
- Animation attributes properties tab;

Following we present how the controls are implemented and how they work.

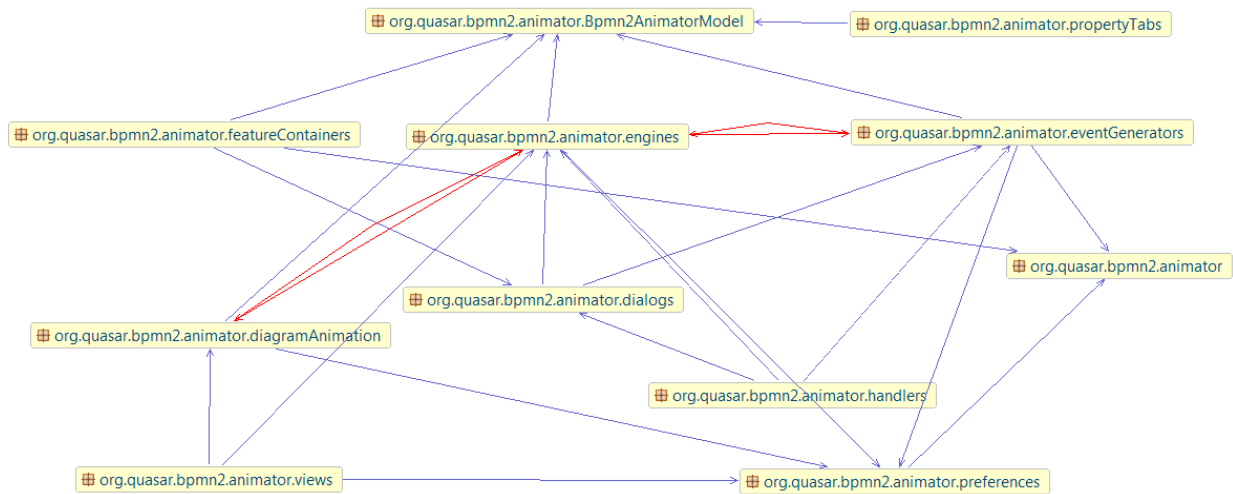


Figure 77 - User controls centered package view

The user controls are arranged in several packages according to their nature. For instance, the “Animation Display Control” view classes are in the views package. The packages that contain classes responsible for the user controls are (see Figure 77):

Package	Package content
org.quasar.bpmn2.animator.views	Animation control view.
org.quasar.bpmn2.animator.preferences	Preferences window.
org.quasar.bpmn2.animator.handlers	Animation execution controls handlers.
org.quasar.bpmn2.animator.propertyTabs	Animation attributes properties tab.
org.quasar.bpmn2.animator.dialogs	Dialogs used in the plug-in like the start animation dialog.

Preferences package dependencies

Figure 77 shows the dependencies that each of the previous packages has on other packages of the plug-in. There are several packages that depend on the preferences package, because several

functionalities are enabled or disabled in the preferences window and therefore it is necessary to access the preferences to see if a certain option is enabled.

Views package dependencies

The views package is dependent on the *diagramAnimation* package because the view tells the *diagramAnimation* classes what are the metrics that should be presented for each BPMN2 element type. The same package also depends on the engines package, because the view calls methods from the *DiagramAnimationEngine* when the options of the view are changed.

Dialogs package dependencies

The dialogs package depends on the *eventGenerators* package and is a dependency for the handlers and the *featureContainers* package. The first is explained by the “event manager” (Figure 47) dialog that communicates directly with the *eventGenerators* to create a new event generator and to give the instruction to read the next event. On the other hand, the handlers depend on the dialogs package, because the start handler displays the start animation dialog and the *featureContainers* depend on the dialogs package because they use the performer dialog that enables the user to choose the performer for an activity.

Handlers package dependencies

The *handlers* package depends on three packages: the dialogs package, the event generators package and the engines package. The first dependency is already explained in the dialogs package dependencies. The dependencies with the *eventGenerators* exist because the *StartAnimationHandler*, which is a class from the *handlers* package, is the one that creates the *eventGenerator* in the start of the animation after the user has chosen the animation type. The last dependency, the one with the engines package, exists because the handlers communicate with the *AnimationEngine* class when an animation execution control is triggered.

PropertyTabs package dependencies

Here the only dependency is with the *Bpmn2AnimatorModel* because in order to show the properties of a model object it is necessary to access it.

Preferences package classes

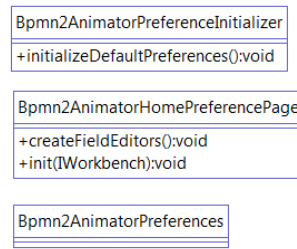


Figure 78 - Preferences package class diagram.

The preferences package contains three classes whose responsibilities and functions are explained in the following tables:

Table 51 - Preferences package classes' description

Class	Description
Bpmn2AnimatorPreferenceInitializer	Class where the preferences default values are defined and stored in the <i>IPreferenceStore</i> .
Bpmn2AnimatorHomePreferencePage	Class that contains the code for the page that is presented in the Eclipse IDE preferences window.
Bpmn2AnimatorPreferences	Class with constants that are used in the preferences page.

Handlers package classes

The handlers' package contains the handlers for the three animation execution controls (start / resume, pause and stop) and contains also the class responsible for the animation speed control.

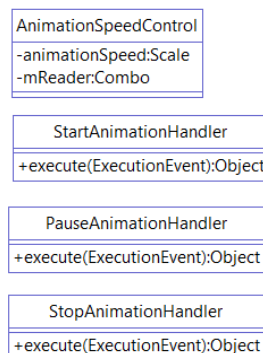


Figure 79 - Handlers package class diagram.

Table 52 - *Handlers* package classes description

Class	Description
<i>AnimationSpeedControl</i>	Class responsible for the speed control.
<i>StartAnimationHandler</i>	Class that implements the logic for the start and also for the resume of the animation.
<i>PauseAnimationHandler</i>	Class that implements the pause animation handler.
<i>StopAnimationHandler</i>	Class that implements the stop animation handler.

Views package classes

This package contains only one class with an internal class. The class contains the code for building the “Animation Display Control” view and also the listeners for each option of the view.

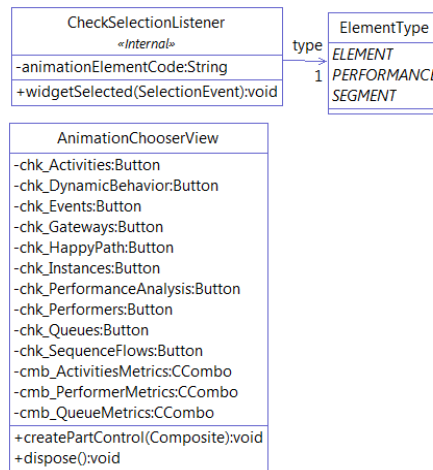


Figure 80 - *Views* package class diagram

Table 53 - Classes descriptions

Class	Description
<i>AnimationChooserView</i>	This class contains the other two classes presented in this table. The class contains the code to build to show the view.
<i>CheckSelectionListener</i>	Class that implements the logic that deals with the events triggered.
<i>ElementType</i>	Enumerator class that defines the possible types used to define the view controls. This type is then used in the logic of the <i>CheckSelectionListener</i> .

6.4.4. Bpmn2Animator event generators

Event generators, produce the events that are sent to the *AnimationEngine* and, in the end, are displayed in the model diagram.

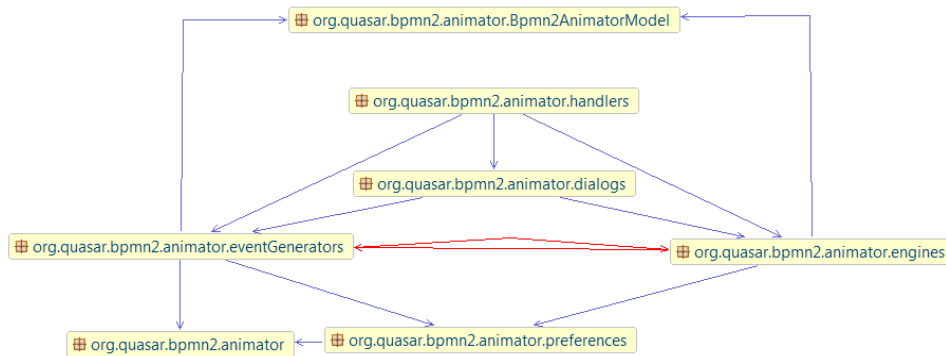
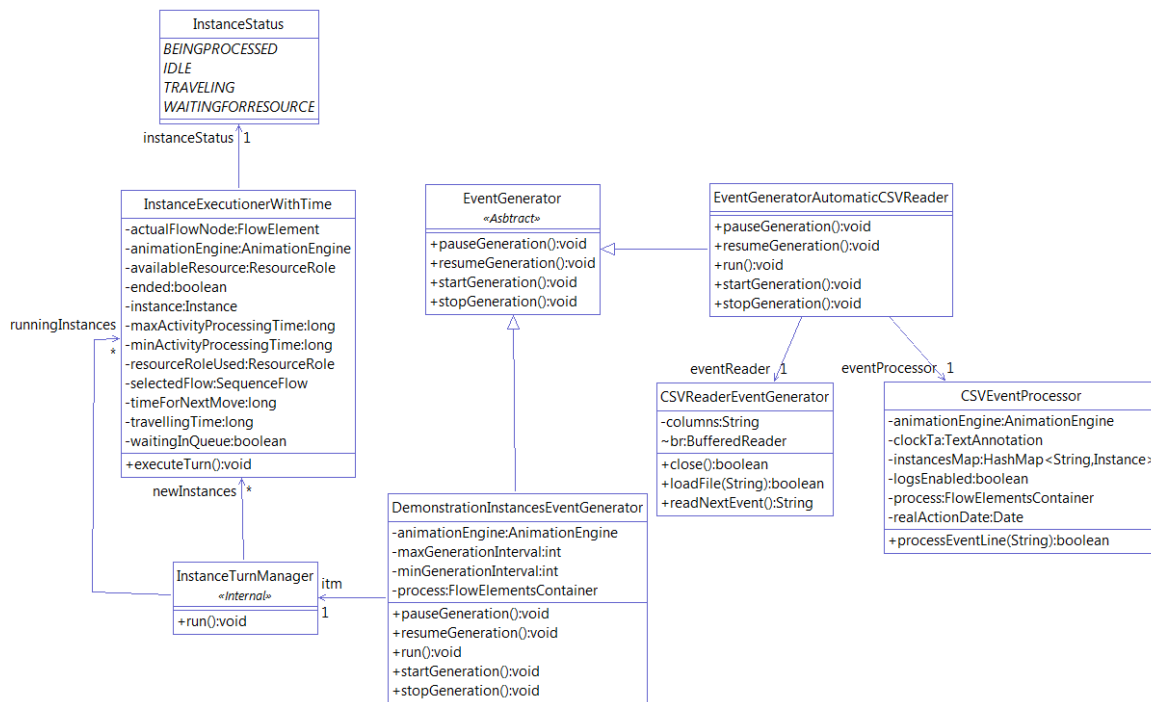


Figure 81 - *EventGenerators* package dependencies diagram.

As shown in Figure 81 the *eventGenerators* package depends on several packages and several other packages depend on it. Some dependencies have already been explained in the previous sections of this chapter, for example, the *eventGenerators* depends on the engine because it send the events to the *AnimationEngine*. On the other hand, the *AnimationEngine* depends on the *eventGenerators* because it controls the execution of the event generators. It depends on the animator package because it uses the plug-in *Activator* class to log the events information in the “Error log” view of the Eclipse IDE. The dependency on the preferences package is explained by the “enable execution logs” functionality that the event generator checks to know if it should log the event.

The existent dependencies with the handlers and dialogs packages were already explained in the previous section. Briefly the first is because the *StartAnimationHandler* instantiates objects from the *eventGenerators* package, whereas the second one is due to the “event manager” dialog that communicates directly with the *eventGenerators* classes.

Figure 82 - *EventGenerators* package class diagram.

There are two types of event generators, which means that exists two classes that extend the *EventGenerator* class as it is possible to see in Figure 82. The first is the *EventGeneratorAutomaticCSVReader* which objective is to generate events to the *AnimationEngine* by reading events from an CSV file, that contains the events information. This class contains one instance of an *CSVReaderEventGenerator* and an instance of an *CSVEventProcessor*, and it is possible to see in the association roles, that the first is used as the event reader, whereas the second one is used as the event processor. The event reader is the object that reads the events from the file. On the other hand, the event processor is the object that interprets the events and sends the event information to the *AnimationEngine* in order to animate the event in the model diagram. The *DemonstrationInstancesEventGenerator* objective is to randomly create instances that will travel trough the process just for demonstartion purposes, choosing the gateways paths according to generated random values. The *DemonstrationInstancesEventGenerator* generates requests for the creation of new instances to the *InstanceTurnManager* randomly. The *InstanceTurnManager* manages the requests for the creation of new instances and also manages the instances that are running. Each running instance is represented by an *InstanceExecutionerWithTime* that contains all the execution information of the instance and calculates the next process step for its instance. The actions are executed with a time interval that is randomly calculated in the previous action.

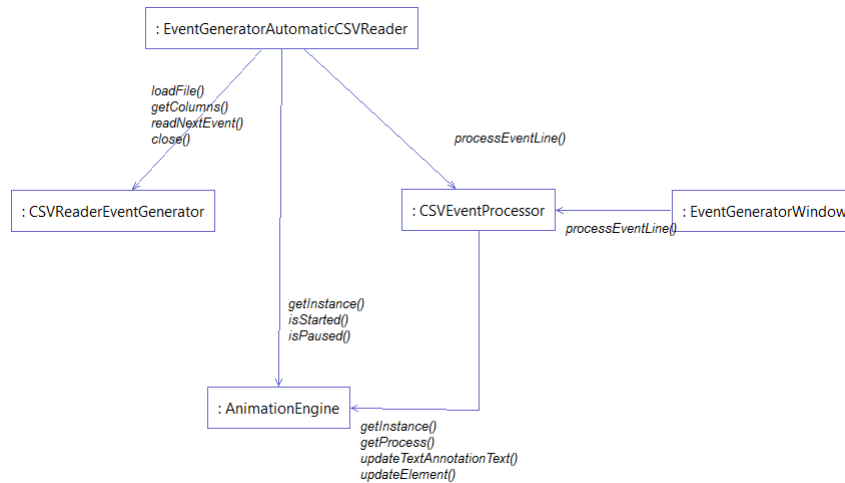
Figure 83 - *EventGenerator* interaction diagram

Figure 83 shows the interactions that exist between the classes that work together with the *EventGeneratorAutomaticCSVReader* to read the events and process them and ultimately send them to the *AnimationEngine*. The *EventGeneratorWindow* interacts directly with the *CSVEventProcessor* when the “Event reader with Event Manager” option is selected and the user uses the event manager dialog (Figure 47) and asks for the next event. If the chosen animation type is “Automatic event reader” the *EventGeneratorWindow* is not used. Instead the *EventGeneratorAutomaticCSVReader* and the *CSVReaderEventGenerator* are used. The *EventGeneratorAutomaticCSVReader* acts as a manager that requests the *CSVReaderEventGenerator* to read the events from the file, by using the method named *readNextEvent* and then sends the event data to the *CSVEventProcessor* through the method *processEventLine*. This is done cyclically, until the end of the file. The *AnimationEngine* is used by the *EventGeneratorAutomaticCSVReader* to obtain a running instance and to check if the animation is paused or stopped, is also used by the *CSVEventProcessor* to send the events information and also to get the process model and a running instance of the engine.

6.5. Summary

In this chapter the Bpmn2Animator prototype was presented. First, the prototype use cases were presented together with illustrations of the user controls. Next, the plug-in development was presented. In this section the objective was to show the Bpmn2Animator dependencies with the other Eclipse IDE plug-ins and how the Bpmn2 Modeler plug-in was extended in order to add the animation functionalities. After it, a package analysis of the prototype was explained in order to provide an understanding on how the prototype works internally. The idea with this chapter is to provide the necessary technical documentation to someone that desires to proceed with our work and use Bpmn2Animator for other research purposes. The other objective of this chapter was to show that the

Bpmn2Animator is a plug-in to be used on top of Bpmn2 Modeler, and that it was not necessary to change the Bpmn2 Modeler in any way to provide the presented functionalities.

Chapter 7

Validation

Contents

7.	Validation	106
7.1.	Introduction	106
7.2.	Business process – Request fulfillment.....	106
7.3.	Data preparation	110
7.4.	Process execution data replay.....	115
7.5.	Summary	128

In this chapter the validation process of Bpmn2Animator prototype is presented. First we present the context of the case study. The process model is presented and the data collection and preparation is explained. Finally the process animation is presented and described.

7. Validation

7.1. Introduction

The Bpmn2Animator plug-in, presented in the previous chapters, was used to animate the model of a real life process by replaying real data extracted from an IT Service Management System. The process used in this case study is the *Request Fulfillment Management* process, as it is currently implemented in ISCTE-IUL University. The process was modeled in BPMN2 using the Bpmn2 Modeler plug-in and model parameterization was performed in order to make the animation possible. The data extraction and processing is explained and the process replay of data gathered from the IT Service Management System is presented together with its values description. In the end some conclusions are drawn.

7.2. Business process – Request fulfillment

In (Taylor et al. 2011) the request fulfillment service operation process is described as “*the process of dealing with Service Requests from the users*”. A Service Request can be seen as “*a generic description for many varying types of demands that are placed upon the IT Department by the users*” which is, in the majority of cases, low risk, frequently occurring and low cost (Taylor, Cannon and Wheeldon 2011). In (Taylor, Cannon and Wheeldon 2011) exists a section called “*Process activities, methods and techniques*” regarding the Request Fulfillment process where the five points presented in the two leftmost columns of Table 54 are described.

Table 54 - Request Fulfillment process activities

Process Activity, method or technique	Description	Model Activities
Menu selection	This activity stands for the creation of the Service Request in the IT Service Management system. Here, the user can select the Service Request type from a service request catalog and input details about it.	Create service request
Financial approval	This activity contemplates the negotiation and approval of the Service Request Cost.	Analyze and classify (Help desk)
Other approval	In the cases where other type of approval is needed it should exist an activity where the approvals can be defined and checked.	Analyze and classify (1 st line) Analyze and classify (2 nd line)
Fulfillment	Activity where the Service Request is fulfilled.	Supply service (Help desk) Supply service (1 st line) Supply service (2 nd line)
Closure	In this activity the Service Desk goes through a closure process in order to check user satisfaction.	Validate service

The rightmost column of Table 54 presents a mapping between the aforementioned points and how they were modeled in Figure 84.

7.2.1. ISCTE-IUL request fulfillment process

The institution process implementation has five roles and basically four different activities in the process, some of which are repeated by the different roles, as presented in Figure 84.

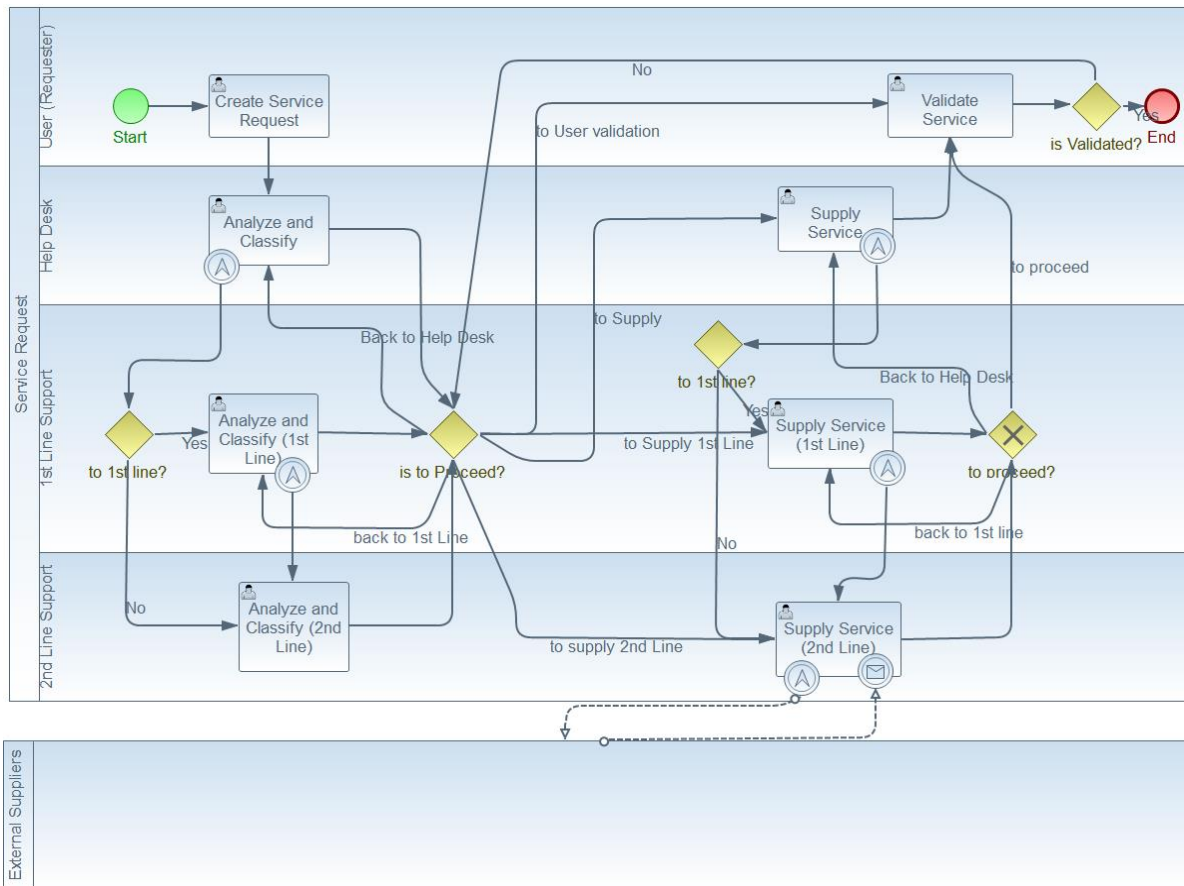


Figure 84 - BPMN model of the request fulfillment process

7.2.2. Process roles

In Figure 84 it is possible to see that the model contains four lanes in the pool named “ISCTE-IUL” and there is another pool without any lane that represents other process role named “External suppliers”. We will now present the process roles, based upon the descriptions provided in (Ferreira 2011).

Table 55 – Request fulfillment process roles

Process role	Description
User	The user role represents the requester, the person who requests the service. The former is able to create a service request and, in the end of the process, validate the service provided to give indication to the provider or whether that the request was well fulfilled or not.
Help Desk	The help desk team is the contact point that the user can use to request a service. The team objective is to ensure that the service requests are treated in

	a consistent manner, minimizing the interruptions of the other lines of support.
1st line support	The 1 st line support team is formed by a group of more specialized technicians than their Help Desk counterparts.
2nd line support	The 2 nd line support team has a greater skillset and tools available than the previous two teams. Since most service requests are fulfilled by the other two teams, the 2 nd line support ends up having more time available to fulfill the requests.
External suppliers	The external suppliers of hardware, software or services can be compared to a 3 rd line support that acts in special occasions where the in house skills are not enough to cope with the request. As presented in Figure 84 their services are usually requested by the elements of the 2 nd line support.

7.2.3. Process activities

The activities that are repeated in different roles (*Analyze and Classify* and *Supply Service*) are grouped and explained only once, since the performed procedures to complete the activity are the same, with the difference that they are performed by different roles.

Create Service Request

When the user cannot satisfy his need autonomously, through the use of the self-service resources provided by the IT Department, he can create a Service Request by using the IT Service Management System web interface available for all users. In order to facilitate the request creation the user has available a Service Request to facilitate the classification of the service being requested. This activity is related with the “Menu Selection” described in Table 54. When this activity is completed, the process advances to the “Analyze and classify” activity to be performed by the Help Desk team.

Analyze and Classify

The help desk team should identify, register and, when necessary, complete the Service Request created by the user (requester) in the IT Service Management Tool web interface. In the case when the help desk team is not capable of performing this activity, the team can escalate the service request to the 1st line support or directly to the 2nd line support. When any of the teams mistakenly escalate the activity to another team, the receiver of the escalation can return the activity back to the first team. From this activity, the process goes to the “Supply Service” activity or, in some cases, when the service is provided directly in the first activity, it can go to the “Validate Service” activity.

Supply Service

In this activity, one of the IT Department teams (Help Desk, 1st line support or 2nd line support) supplies the service that was requested by the user (requester). This activity also supports escalation, just like the “Analyze and Classify” activity and is also possible, if there is a wrong escalation, to send the activity back to the team that escalated the request. Besides these possibilities, this activity has yet another particularity. The 2nd line support is allowed to escalate the service request to external suppliers, when there is no possibility for the service to be provided by any of the IT Department teams. After this activity is completed (i.e. after the service is supplied) the process advances to the “Validate Service” activity.

Validate Service

The “Validate Service” activity occurs in the end of the process, after the request has been fulfilled. This activity allows the user to confirm and evaluate the fulfillment of the service request. This task concludes the process or, if the service has not been supplied as expected by the user, the process goes back to the “Supply Service” activity corresponding to the role that supplied the service.

7.3. Data preparation

This section describes the data collection and pre-processing required for animating the Request Fulfilment Management process, presented in the previous section. First, the data collection process is explained together with a presentation of the data in its original form. Then, the data transformation needed in order to use the data in the Bpmn2Animator plug-in is presented, together with a presentation of an Eclipse IDE plug-in that was developed in order to automate the data preparation. It was also necessary to parameterize the process model so the Bpmn2Animator could animate it. The process model parameterization work is presented in chapter A.D.

7.3.1. Data collection

As mentioned before, all the data related to the Request Fulfilment Management process is stored in the IT Service Management System of the institution. From the system it is possible to collect all data related with the execution of the process, by using its web interface. In order to do so, it is necessary to access the “Operation” menu, then select the option “Actions” and finally select the option “All Actions”. In this web page it is possible to explore the available data by using the “Filter” and the “View” options.

The screenshot shows a web interface for an IT Management System. At the top, there are logos for ISCTE and IUL iAjuda. A navigation menu includes options like Home, My Department, Discovery, Usage, SNMP, Asset Management, Transition, Design, Strategy, Extended CMDB, Continual Improvement, Administration, Integration, and Logout. A 'Quick Call [ALT+Q]' widget is visible with status indicators for 'Late' and 'To Do' counts. The main content area displays a table of actions under the 'Actions' menu. The table has columns for 'Number', 'Type', 'Description', 'Creation Date', and 'Group'. The data rows show various actions such as 'Operation Processing', 'Send Email', 'Reallocation', and 'New RFC', with descriptions in Portuguese and creation dates ranging from 28/09/2013 to 27/09/2013. A sidebar on the left contains a navigation menu and an 'INFORMATION' section with user details for Marco André Marques Roque.

Number	Type	Description	Creation Date	Group
S130928_000004	Operation Processing	Análise e Classificação	28/09/2013 09:58:11	Helpdesk
S130928_000004	Send Email	User notification	28/09/2013 09:58:11	-
S130928_000004	Reallocation	Reallocation	28/09/2013 09:58:11	Helpdesk
S130928_000004	Operation Processing	Fornecer serviço	28/09/2013 09:58:11	1st Line Support
S130928_000004	Send Email	Send Email	28/09/2013 09:58:11	-
S130928_000002	Reallocation	Reallocation	28/09/2013 09:37:51	Helpdesk
S130928_000002	Operation Processing	Fornecer serviço	28/09/2013 09:37:51	Helpdesk
S130928_000002	Escalation	Fornecer serviço	28/09/2013 09:37:51	Helpdesk
S130928_000002	Operation Processing	Análise e Classificação	28/09/2013 09:37:51	Helpdesk
S130928_000002	Send Email	User notification	28/09/2013 09:37:51	-
S130928_000001	Send Email	Send Email	28/09/2013 09:35:07	-
S130928_000001	Reallocation	Reallocation	28/09/2013 09:35:07	Helpdesk
S130928_000001	Operation Processing	Fornecer serviço	28/09/2013 09:35:07	1st Line Support
S130928_000001	Operation Processing	Análise e Classificação	28/09/2013 09:35:07	Helpdesk
S130928_000001	Send Email	User notification	28/09/2013 09:35:07	-
S130927_000062	New RFC	Creation	27/09/2013 18:10:28	Helpdesk
S130927_000062	Operation Processing	Fornecer serviço	27/09/2013 18:10:28	Suporte a Auditórios
S130927_000062	Front Office Validation with Notation	Confirmação e avaliação do fornecimento do serviço	27/09/2013 18:10:28	-
S130927_000062	Send Email	Send Email	27/09/2013 18:10:28	-
S130927_000061	New RFC	Creation	27/09/2013 18:08:12	Helpdesk
S130927_000061	Operation Processing	Fornecer serviço	27/09/2013 18:08:12	Suporte a Auditórios
S130927_000061	Front Office Validation with Notation	Confirmação e avaliação do fornecimento do serviço	27/09/2013 18:08:12	-
S130927_000061	Send Email	Send Email	27/09/2013 18:08:12	-
S130927_000060	Front Office Validation with Notation	Confirmação e avaliação do fornecimento do serviço	27/09/2013 18:06:24	-
S130927_000060	Send Email	Send Email	27/09/2013 18:06:24	-
S130927_000060	New RFC	Creation	27/09/2013 18:06:24	Helpdesk
S130927_000060	Operation Processing	Fornecer serviço	27/09/2013 18:06:24	Suporte a Auditórios
S130927_000059	New RFC	Creation	27/09/2013 18:04:15	Helpdesk
S130927_000059	Operation Processing	Fornecer serviço	27/09/2013 18:04:15	Suporte a Auditórios
S130927_000059	Front Office Validation with Notation	Confirmação e avaliação do fornecimento do serviço	27/09/2013 18:04:15	-
S130927_000059	Send Email	Send Email	27/09/2013 18:04:15	-

Figure 85 - IT Management System data export web page (Operation -> Actions -> All Actions)

The filter used to retrieve the required information for the case study, selects only actions created and finished between September 2nd and September 30th 2013. This month was chosen because it is when the school year begins and when all student subscriptions are made. This is a critical time of the school calendar for the IT department that needs to respond timely to all kinds of service requests such as: problems with inscriptions; IT infrastructure with preparations for the first classes; difficulties in accessing the institution web portals. The fact that this month is so critical and so stressful to the IT department makes its study interesting, because it will be possible, through process model animation, to see how the IT department team reacts to this extra workload. In order to see only the actions created and finished in September, the “Creation Date” and the “Real End Date” filters are set to be between 02/09/2013 and 31/09/2013.

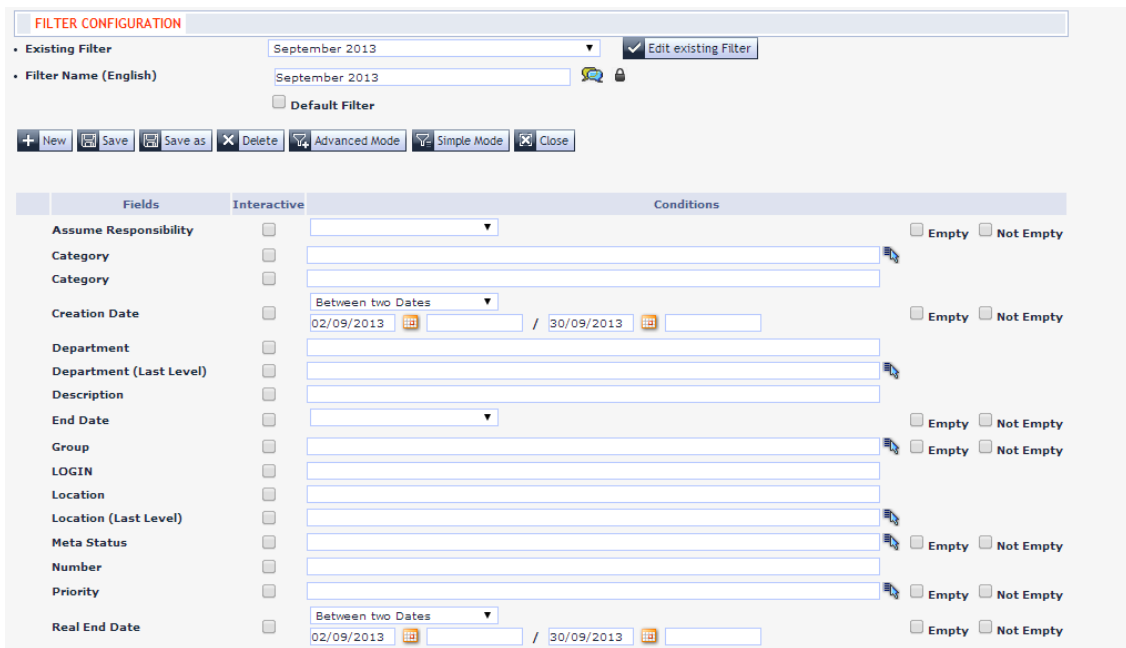


Figure 86 - IT Service Management System - Filter web interface

Complementary to the filter, the view is used to establish which columns are retrieved by the IT Service Management System. For the present study the relevant columns are the ones presented in Table 56.

Table 56 - Columns selected in the view

Column	Description
Number	The id of the Service Request or Incident
Type	Action type (see Table 64)
Description	Action description
Creation Date	The ticket creation date, it is equal for all actions of the same ticket
Group	The group (e.g. Help Desk) that performed the action (see Table 66)
Real End Date	The action end date
Scheduled End Date	Deadline for the Service Level Agreement violation
End Date	The end date of the ticket
Topic	The topic selected from the catalog that identifies the type of service request or incident

Finally, after selecting the filter and view, it is possible to extract all the actions to a CSV file. Unfortunately, exporting the data this way means that not only the service request data is retrieved, but also the incident management data is retrieved meaning that some additional filtering is required as described in the next section.

7.3.2. Data preparation

The event generator, which is used to read the process execution data, needs more attributes than the ones that exist in the extracted data. On the other end, the collected events contain data that is not relevant for the case study. This means that, in order to prepare the data for the Bpmn2Animator it is necessary to exclude the unnecessary actions and add some attributes that will be used in the replay of the process execution data. In order to facilitate the data preparation task we developed an Eclipse IDE plug-in, named CSV Analyzer, that reads the exported data, processes it and generates a new file with the actions ready to be executed in the Bpmn2Animator.

However, before using the CSV Analyzer, it is necessary to add the columns “Group_level”, “sortHelpCol” and “sortHelper”. The description for each column is presented in Table 57. After adding the columns it is necessary to sort the actions by Number, Real End Date and sortHelper column like presented in Figure 87. The “sortHelper” column is filled with the sort level that corresponds to the action type. The sort level for each action type can be seen in Table 63.

Column	Sort On	Order
Sort by	Number	Values
Then by	Real End Date	Values
Then by	sortHelper	Values

Figure 87 - Sorting criteria to prepare the file for CSV Analyzer plug-in

The remaining columns presented in Table 57 are added automatically by the CSV Analyzer (to know more about CSV Analyzer see section A.C.1).

Table 57 - Columns added in the first stage of the data preparation

Group_level	Description	Indicates the level of the group from the column “Group”.
	Origin	Calculated based on a table exported from the IT Service Management System ("Directory" -> "Groups"), which contains the group level for each group present in the "Group" column. The Groups table can be seen in Table 66.
	Used for	Used by the event generator of the Bpmn2Animator plug-in in order to find which path an instance must travel in the gateways.
sortHelpCol	Description	Concatenation of the “Type” column with the “Description” column.
	Origin	Example of the formula: CONCATENATE(B2;" ";C2)
	Used for	It is used to calculate the "sortHelper" column value.
sortHelper	Description	Value used to sort of the actions.
	Origin	Lookup of the sortHelpCol value in a table that contains the sort order as can be seen in Table 63.
	Used for	It is used to sort the actions, so the actions have the correct order to be read by the CSV Analyzer plug-in.
Escalation_Group	Description	The group that will realize the next action. This is used in all

		cases: “Escalation”, “Transfer Error” or “Operation Processing”.
	Origin	Calculated by the "CSV Analyzer". This column is filled with the "group_level" value of the group that will perform the next activity.
	Used for	Used by the event generator of the Bpmn2Animator plug-in in order to find which path an instance must travel in the gateways.
is_Finished	Description	Indicates if the action is the last of the process instance.
	Origin	Calculated by the "CSV Analyzer". This column is filled when the action is the last one of a process instance.
	Used for	It is used by the Bpmn2Animator event generator in order to understand if an instance should travel to the end event or, instead, go back to a previous activity.
On_hold	Description	Indicates that the action will be on hold, which means that it will not count for queue time metrics.
	Origin	Calculated by the "CSV Analyzer". This column is filled when there is an "on hold" type action, after the action that is being processed.
	Used for	Used by the event generator of the Bpmn2Animator plug-in in order to understand if it should take into account the time between the present action and the next one.
next_activity	Description	Indicates the next action that will be executed.
	Origin	Calculated by the "CSV Analyzer". This column is filled with the type of the next action.
	Used for	Used by the event generator of the Bpmn2Animator plug-in in order to find which path an instance must travel in the gateways.

The final step to complete the data preparation is to use the CSV Analyzer plug-in. In order to do so, it is necessary to install it in Eclipse and then select the menu with the plug-in name, “CSV Analyzer”. Afterwards, a window will be displayed showing two options, as presented in Figure 88. The second option, “CSV Builder”, should be selected. In the end a new file will be created in the same folder of the original CSV file. This new CSV file has to be sorted by “Real End Date” before being used in the Bpmn2Animator, so all actions are sorted by their execution time from older to newer.

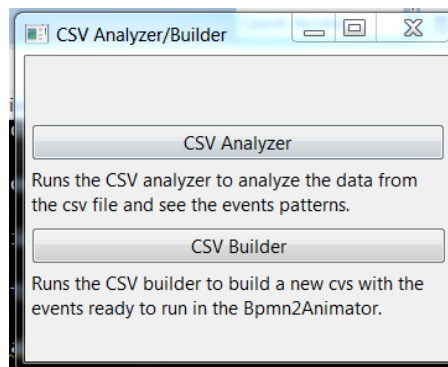


Figure 88 - CSV Analyzer options window

7.4. Process execution data replay

In this section we present a brief analysis of the process prepared execution data regarding September month of 2013. Afterwards, we analyze three days, where activity spikes were found. Three figures of the animated process model are presented for each analyzed day. In each figure the model elements are presented with different colors that depend on the chosen metric that the color represents. Each figure will now be analyzed in detail. The process elements execution data that is stored by the Bpmn2Animator during the process replay is presented in section A.E of the Annexes.

7.4.1. Service request fulfilment process replay

Figure 89 shows that 3 peaks in the number of realized actions occurred in the month of September. The peaks occurred in the 11th, 19th and 26th of September, with the two first days coinciding with peaks in the number of created service requests. On the other hand, the peak on the number of realized actions on the 26th of September is not accompanied by a peak of the number of created service requests. This happens because the service requests were filtered by creation date and conclusion date before the 30th of September, which means that the service requests that were created, but were not concluded before the mentioned date, are not included in the collected data.

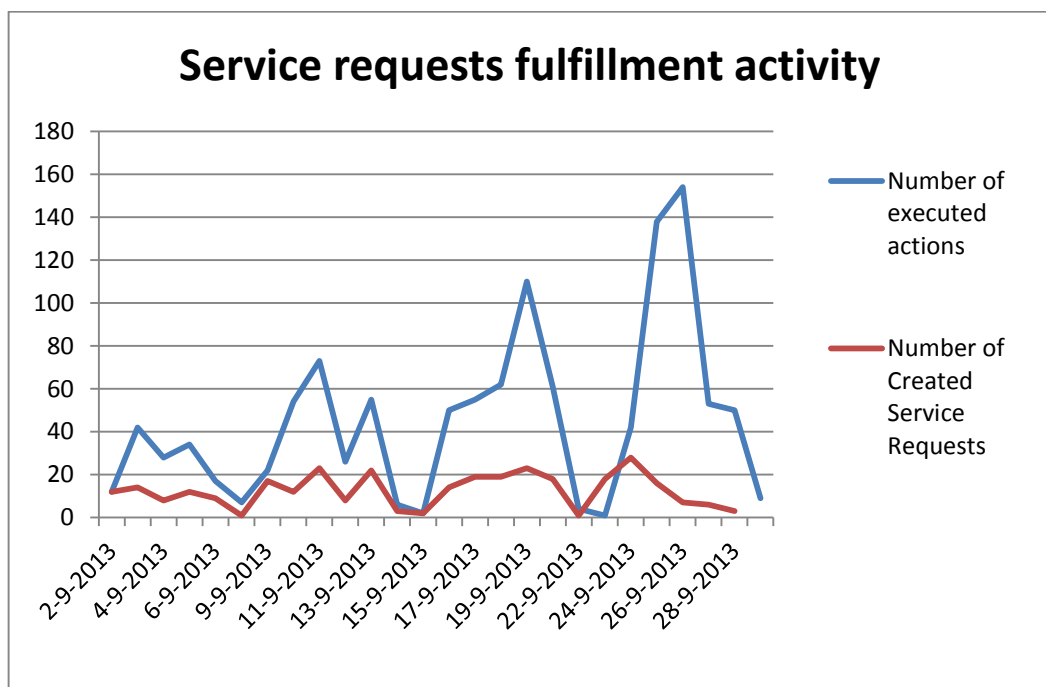


Figure 89 - Service requests fulfilment process activity graphic

This information helps to understand which days are most interesting to analyze during the replay of the collected data with the Bpmn2Animator plug-in. The next section describes a replay of the

collected data with the help of figures gathered from the plug-in for the aforementioned peak days, with the intent of understanding the state of the process in those moments.

7.4.2. 11th of September, 2013

Figure 90 presents the state of the process in the eleventh day of September. The unique element from the dynamic behavior animation view that can be visualized in this figure is the number of instances waiting on the queues, which is represented by the queues size together with a number. The queues that contain more instances in the end of the day are the queues of the activities: “Analyze and Classify”, “Supply Service (1st line)” and “Supply Service (2nd line)”. The “Validate Service” activity has the highest number of instances waiting on its queue, but since it is an activity performed by the requester, it is not relevant for our analysis.

Path Load Analysis

Observing the Sequence Flows color and the model elements border line color it is possible to understand which path is more frequently used. The “Create Service Request” and “Analyze and Classify” activities are the ones that were more frequently performed as it is possible to see by observing its incoming and outgoing Sequence Flows that are red, as well as their border line. By analyzing the process from start to end it is possible to verify that the most frequent path is the one that passes through the activity “Supply Service (1st line)” and then goes directly to the “Validate Service” activity. On the other hand, the activity “Analyze and Classify (1st line)” stands out because it was never executed as it is possible to verify by its border line color and its surrounding Sequence Flows color as well.

Escalations

Escalations are represented by the border events that exist in the “Analyze and Classify” and “Supply Service” activity type. As it is possible to observe in Figure 90, all of them have a blue or gray border line, which means that they did not exceed 25% of the total of instances. As presented in Table 77 of the Annexes, the activities that have the higher number of escalations are the “Supply Service (1st line)” and “Supply Service”, with 8 and 6 instances respectively.

Activities

The activities colors in Figure 90 represent the number of processed instances like its border line color. With this configuration it is possible to easily understand which activities are performed more. In this case the activities are: “Create Service Request”, “Analyze and Classify” and “Supply Service”. In Figure 91 the activities appear colored according to the number of instances whose SLA was

violated. In the figure, the activity “Analyze and Classify” is presented with red color meaning that it was the activity with the higher number of SLA violations.

Queues

Just like activities, the queues are colored depending on the number of instances that entered the queue. Figure 90 shows that the queues that had the highest number of instances waiting on them are the “Create Service”, “Analyze and Classify”, followed by the “Validate Service” and the “Service Supply” activities. Figure 91 shows the queues colored according to the total of minutes that the instances waited in the queue and it is possible to see that the two queues with the highest value are the “Analyze and Classify” and the “Supply Service (1st line)” activities. Figure 92 shows the queues colored according to its average wait time per instance. In this case, the activities that show the highest results are the “Supply Service (1st line)”, “Analyze and Classify (2nd line)” followed by the “Analyze and classify” presented with yellow.

Performers

In Figure 90, just like the activities, performers are colored regarding the number of processed instances. Therefore, the performers have the same color as the activities. In Figure 91 the performers appear colored based on its own total processing time. It was stipulated that each performer takes 10 minutes to process each instance, because the data collected from the IT Service Management System does not allow us to determine how long each action took to be performed. Using the 10 minutes processing time for all performers means that this value will only vary according to the number of processed instances. The most active performer is the Help Desk performer, which performs the “Analyze and Classify” activity, followed by the other Help Desk performer that performs the “Supply Service” activity.

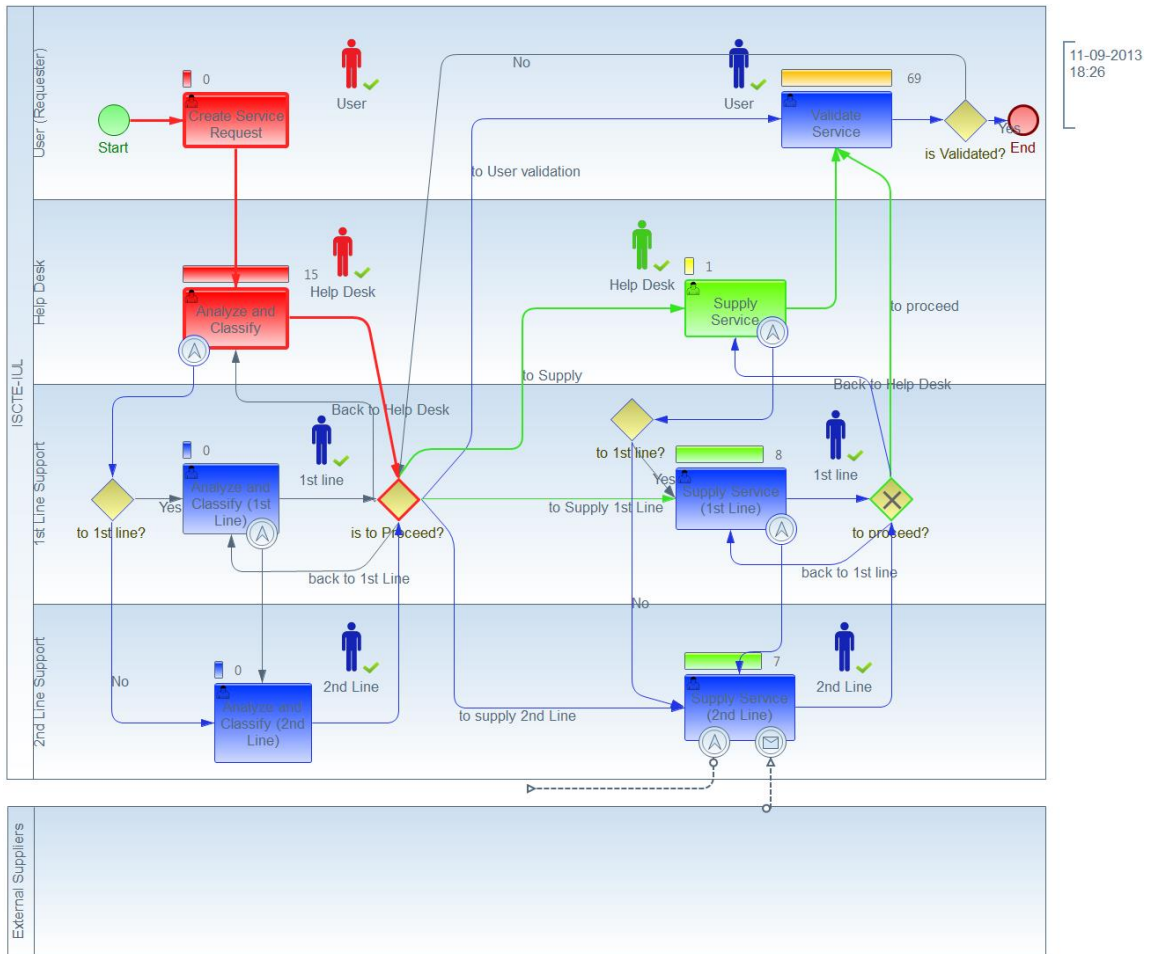


Figure 90 - Process status 11th September (presented metrics: Activities - num. of proc. instances; Performers - num. of proc. instances; Queues - num. instances that waited in the queue)

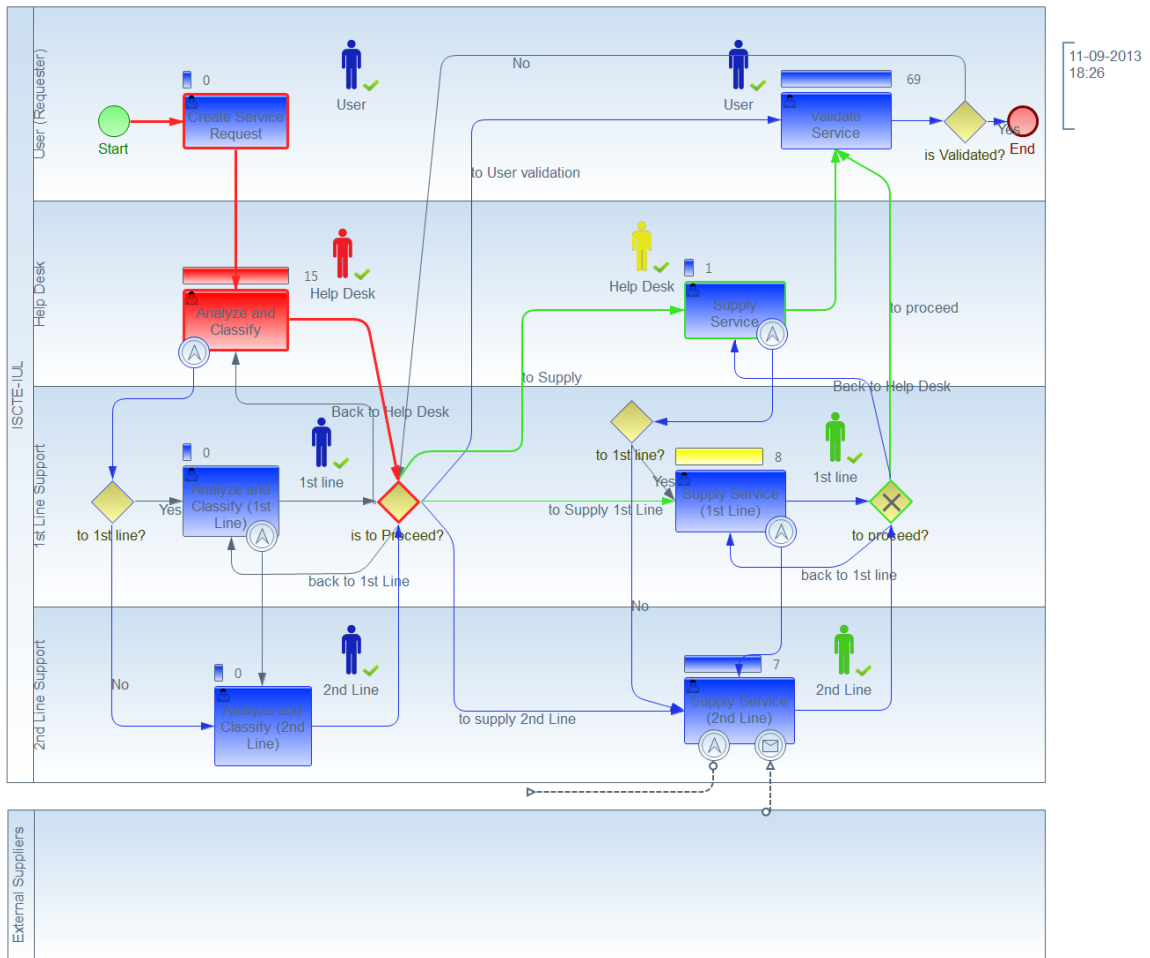


Figure 91 - Process status 11th September (presented metrics: Activities - num. of SLA violations; Performers - total of proc. time; Queues - wait time)

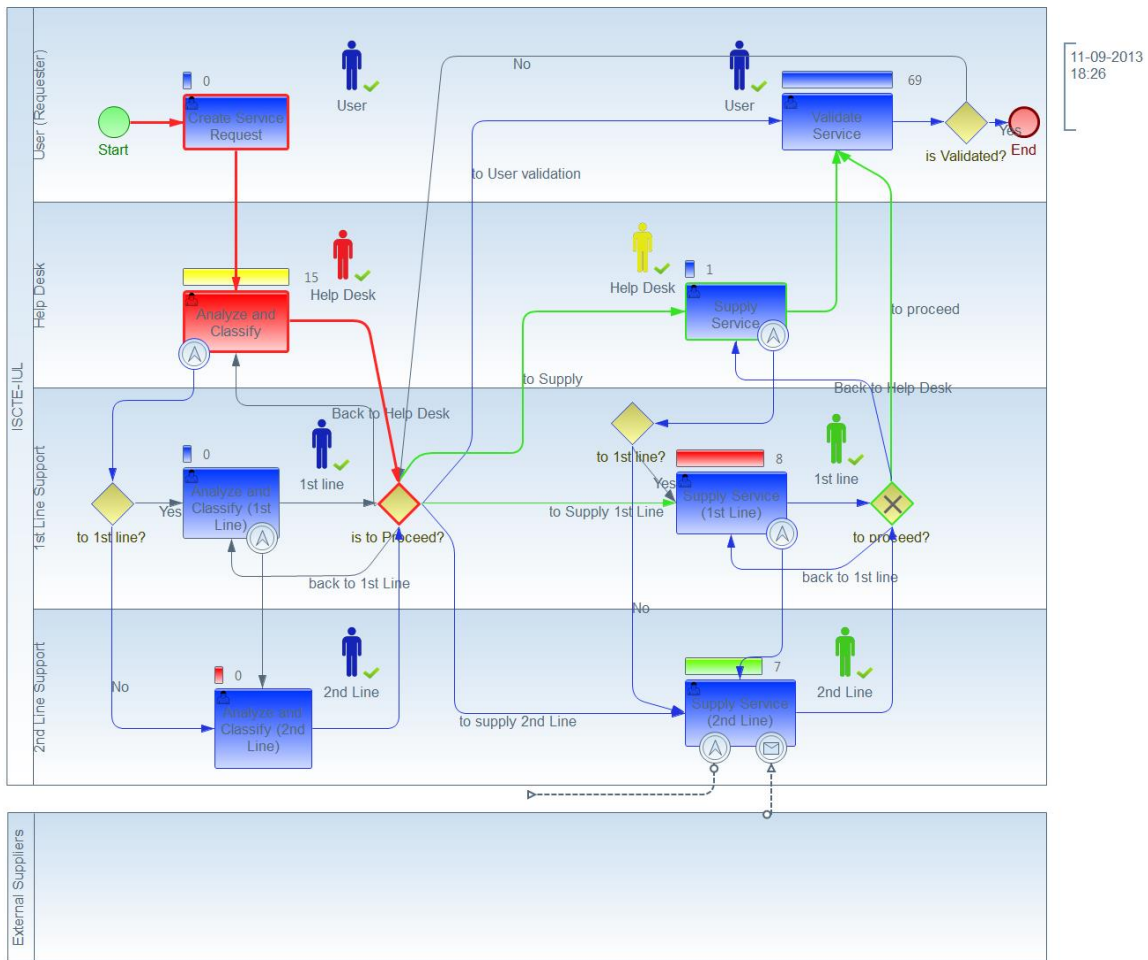


Figure 92 - Process status 11th September (presented metrics: Activities - num. of SLA violations; Performers - total of proc. time; Queues - Avg. wait time per instance)

7.4.3. 19th of September, 2013

Path Load Analysis

The Path Load Analysis view does not present many changes from day 11 to day 19. The only visible change is the evolution of the “Supply Service (1st line)” that is now presented with a green border line, as well as its outgoing Sequence Flow.

Escalations

All activities boundary events remain with the same border line color, meaning that the number of escalations remains less than 20% of the total number of created instances. The data presented in Table 84 shows that the activities with the higher number of escalations remain the “Supply Service” and “Supply Service (1st line)” with 13 and 10 escalations respectively.

Activities

The evolution on the number of processed instances for each activity can be seen through the comparison of Figure 90 with Figure 93. The only change that Figure 93 shows is the “Supply Service (1st line)” activity that is now presented with the green color, similar to what happens to its border line and was already mentioned in the path load analysis before. “Analyze and Classify” remains the activity with the higher number of SLA violations (51). “Supply Service (1st line)” activity violated the SLA for 7 instances and the other activities did not violate any SLA.

Queues

The activities color animation remains the same as it is possible to see comparing Figure 90 with Figure 93, except for the “Validate Service” activity queue which color changed from orange to red. The activities with the highest total of wait time presented in Figure 91, with the colors red and yellow, are also presented with the same color in Figure 94. However, the queues of the activities “Validate Service” and “Supply Service (2nd line)” are colored with the green color in Figure 94, meaning that its percentage regarding the maximum total of wait time has increased. Comparing Figure 92 with Figure 95 it is possible to see that the “Supply Service (2nd line)” activity queue has changed its color from green to yellow and the “Validate Service” activity queue changed its color from blue to green, meaning that the average wait time per instance increased for both queues.

Performers

As it was already mentioned, the number of processed instances has increased in percentage for the activity “Supply Service (1st line)”. As presented in Figure 93, the performer of that activity has also changed its color from blue to green. Figure 94 shows the performers colored according to the total of processing time and, by comparing this with Figure 90, it is possible to see that the performer that performs the “Supply Service (2nd line)” is now colored in blue, meaning that he processed less than 20 percent of the total of created instances.

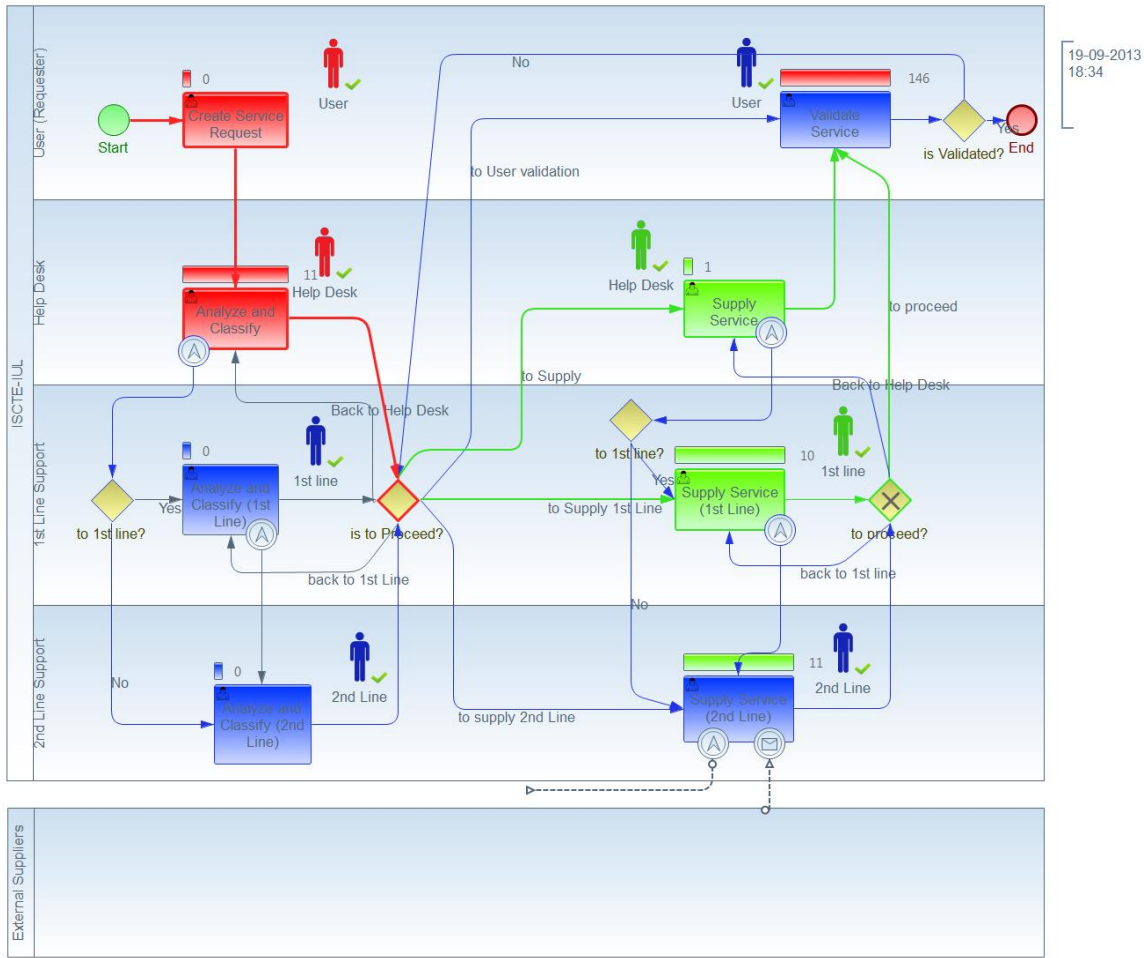


Figure 93 - Process status 19th September (presented metrics: Activities - num. of proc. instances; Performers - num. of proc. instances; Queues - num. of instances that waited in the queue)

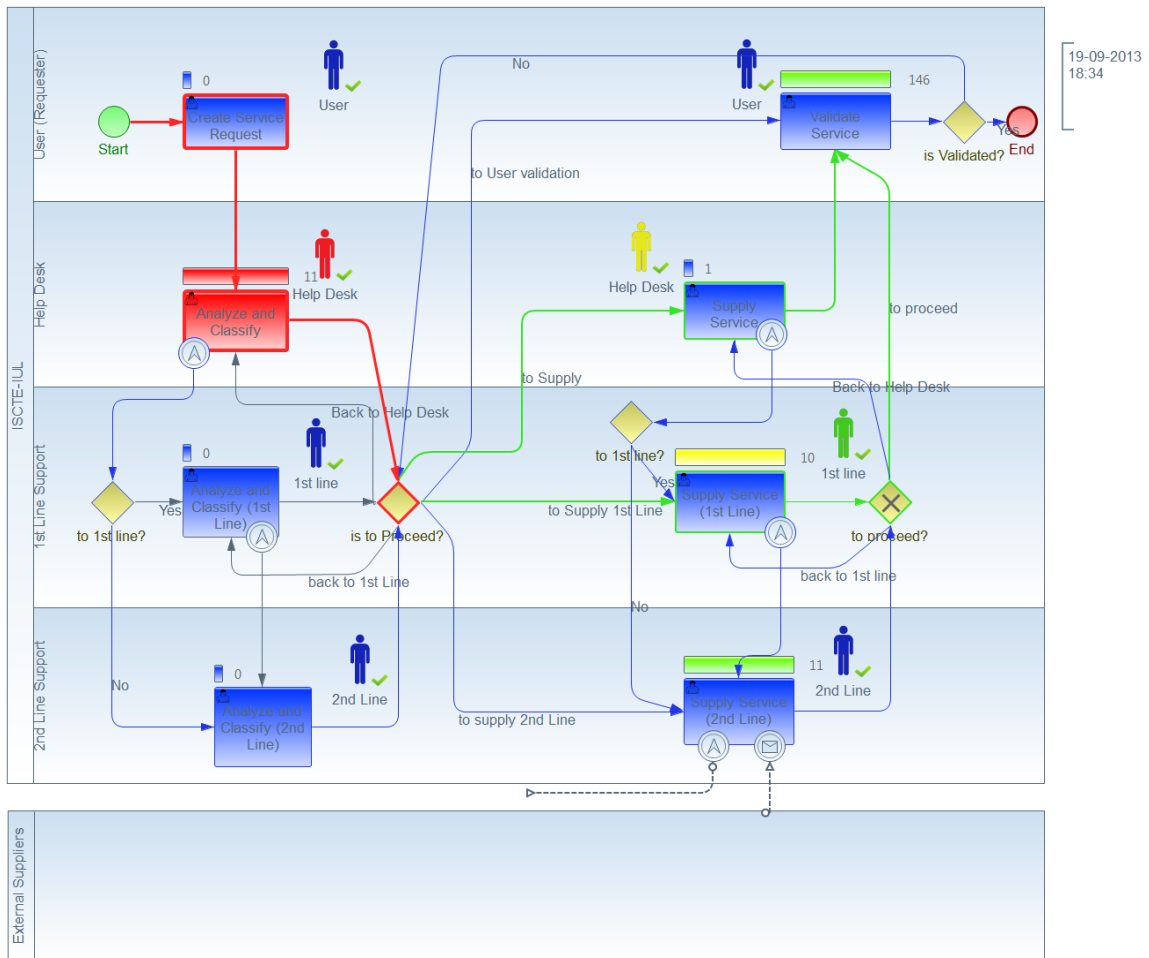


Figure 94 - Process status 19th September (presented metrics: Activities - num. of SLA violations; Performers - total of proc. time; Queues - total of wait time)

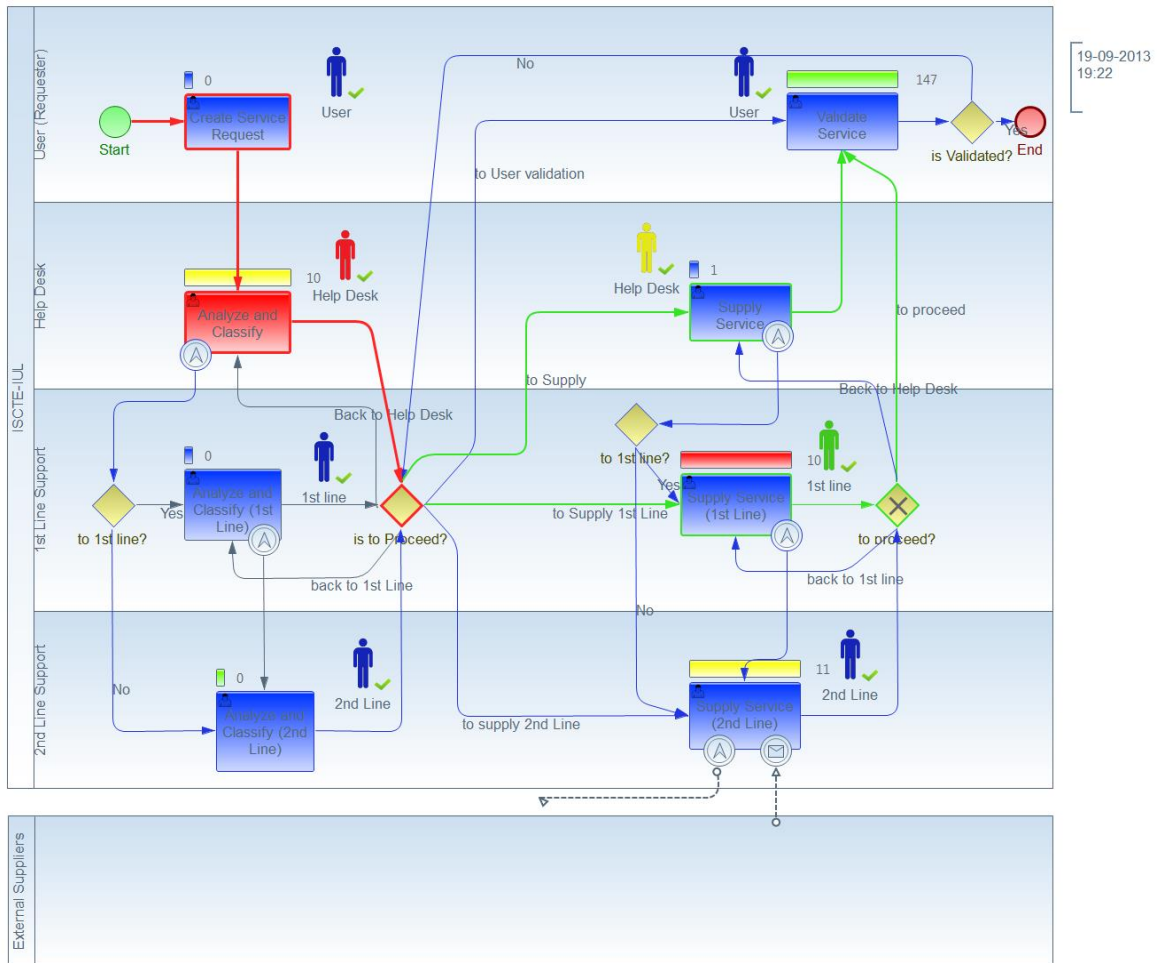


Figure 95 - Process status 19th September (presented metrics: Activities - num. of SLA violations; Performers - total of proc. time; Queues - Avg. wait time per instance)

7.4.4. 26th of September, 2013

Path load analysis

The Path Load Analysis view results changed from Figure 93 to Figure 96. The main change is the color of the ingoing Sequence Flow of “Supply Service” activity, as well as the border line of the activity that changed from green to yellow. Also, the Sequence Flow that goes from the gateway “to proceed” to the “Validate Service” activity has changed its color from green to yellow. These two changes indicate that more instances traveled through both paths. The fact that the “Supply Service” activity outgoing Sequence Flow color has not changed to yellow, indicates that some escalations occurred in the “Supply Service” activity.

Escalations

In terms of animation, everything remains the same. There was no change in the border line of the activity boundary events. By analyzing the data presented in Table 89 it is possible to see that “Supply Service” and “Supply Service (1st line)” remain the activities with the higher number of escalations and that “Analyze and Classify” escalated three instances from 19 to 26 of September.

Activities

Figure 96 shows that “Supply Service” activity has processed a higher percentage of instances changing its color from green to yellow. The animation based on the number of SLA violations shows that the activity with the higher number of SLA violations remains the same and also shows, that the percentage of SLA violations has increased for the “Supply Service (1st line)” activity.

Queues

Comparing Figure 96 with Figure 93, it is possible to see that the “Supply Service (2nd line)” activity queue has changed its color to blue and that the “Validate Service” activity queue has changed to red. The animation results for the total wait time and for the average wait time per instance are different from the previous results, due to the long time that the instances took to be processed in the “Validate Service” activity. This results in a greater value for both metrics, which causes then a change in the colors of the queues when presenting the two metrics, because the color is based on the registered maximum for the metric. Nevertheless, Figure 98 shows that the activity with the second highest average wait time is the “Supply Service (1st line)” activity. The data presented in Table 86 shows that the “Analyze and Classify” and “Supply Service (1st line)” activities are still the ones that have the higher total of wait time. The “Supply Service (1st line)” activity is still the one with the higher average wait time per instance. However the “Supply Service (2nd line)” activity has exceeded the average wait time of the “Analyze and Classify” activity, presenting now the third higher average wait time.

Performers

There were no changes in the color of the Performers for the total of processing time. Regarding the number of processed instances, the changes follow the same evolution as the activities.

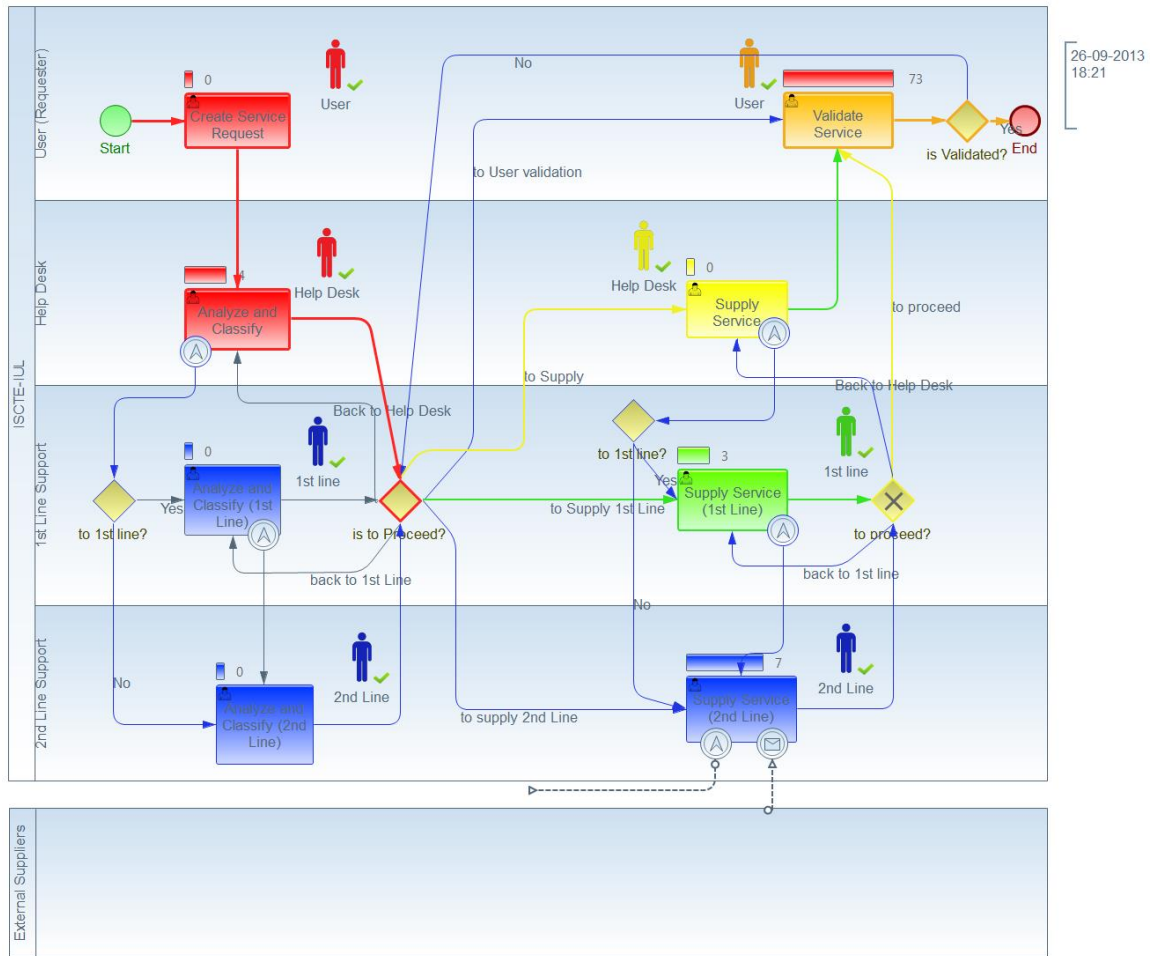


Figure 96 - Process status 26th September (presented metrics: Activities - num. of proc. instances; Performers - num. of proc. instances; Queues - num. of instances that waited in the queue)

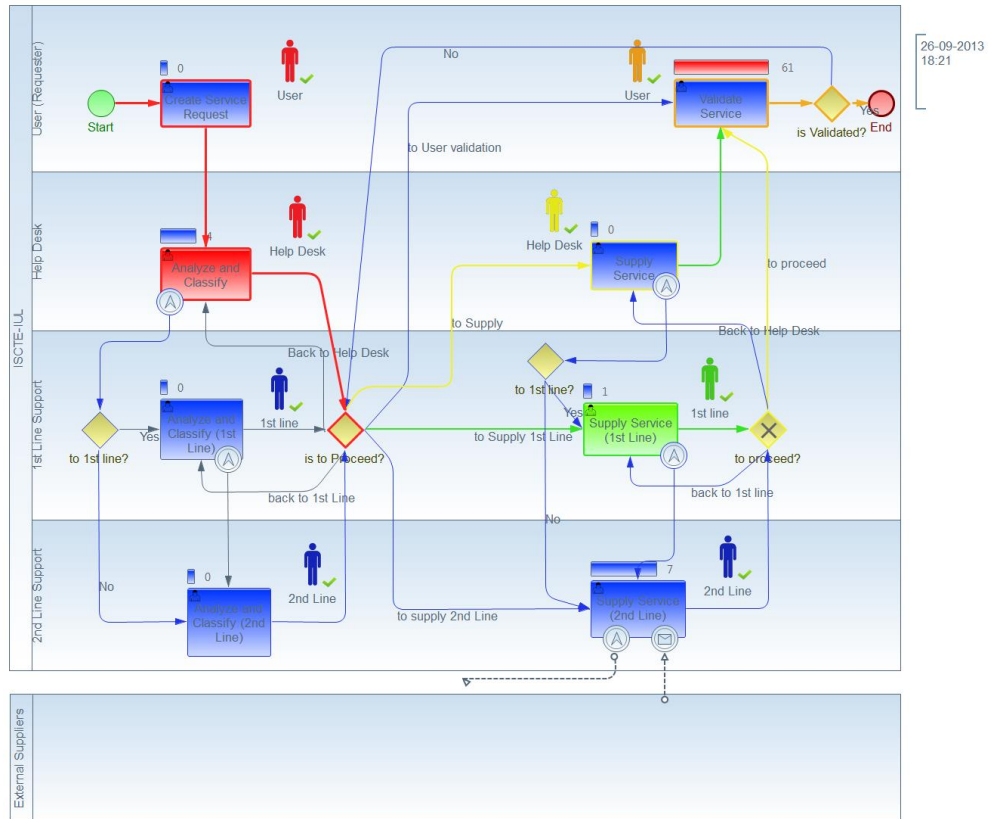


Figure 97 - Process status 26th September (presented metrics: Activities - num. of SLA violations; Performers - total of proc. time; Queues - total of wait time)

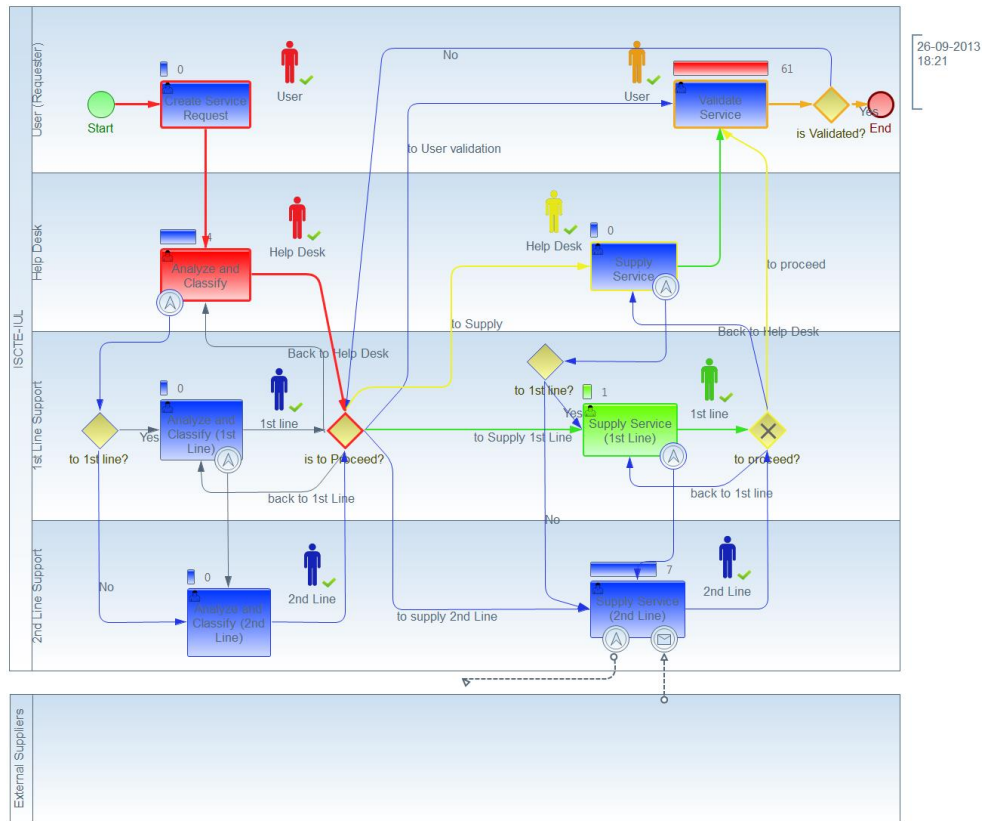


Figure 98 - Process status 26th September (presented metrics: Activities – num. of SLA violations; Performers – total of proc. time; Queues – Avg. wait time per instance)

7.5. Summary

The replay of the process execution data allowed the visualization of the evolution of the model elements during the month of September. The three days, when a peak of performed actions was verified, were analyzed in detail, as well as the evolution of the process elements from one analyzed day to the other.

Chapter 8

Conclusions and future work

Contents

8. Conclusions and future work.....	130
8.1. Conclusions	130
8.2. Future work	132

In this chapter we present our research work conclusions and future work.

8. Conclusions and future work

8.1. Conclusions

The work presented in this dissertation was focused essentially in the animation of business process models in the BPMN2.0 notation. In order to propose the animation for the notation it was necessary to evaluate the state of the art of business processes models animation.

Taxonomy for process animation features

To facilitate the task of evaluating the state of the art we developed and proposed a taxonomy that can be used to evaluate the animation and customization capabilities of the tools that support processes animation. This taxonomy was then applied to evaluate a sample of 6 tools that fulfilled our three requirements for the tool sample selection: Relevance, diversity and availability.

Open source project survey

One of our objectives was the development of a prototype that would implement the animation concepts for the BPMN2.0 proposed in our work. This prototype would have to be developed on top of another open-source BPMN2.0 modeler tool that we could use as a basis of development to achieve our goal. In order to choose the most appropriate tool we established an evaluation criteria that was used to filter the available open-source tools. Then, we developed a fit for purpose evaluation where we evaluated characteristics like API availability and source code understandability. This part consists also in a contribution of the dissertation since it can be used to select open-source tools with other purposes, being only necessary to change the requirements list by one that suites the needs of the evaluator.

BPMN 2.0 elements animation

The main contribution of the developed work is the proposal for the animation of the BPMN2.0 elements. We propose new graphical elements in order to present Resources, Queues and process instances during process animation. We propose also two types of animation 1) border line thickness and color manipulated together and 2) the color fill of the notation elements. This work took into account all BPMN2.0 elements display rules and respected all of them.

Prototype development and application

Other contribution of our work is the prototype that implements the proposed animation for the BPMN2.0 elements. The internals of this prototype are presented and explained in our dissertation which can be useful for anyone that desires to continue our work or just use it to apply it in other realities. The prototype was used to replay execution data from the service request fulfilment process

implemented in ISCTE-IUL. The usage of our plug-in in a real world case study shows that it can be used to analyze business process execution and draw conclusions.

Bpmn2Animator support for process animation

Finally, by evaluating our own plug-in with our proposed taxonomy to evaluate the tool support for process animation we reached the results presented in Table 58.

Table 58 - Bpmn2Animator evaluation results

	Queue	Resource	Process instance	Activity	Sequence flow	Attributes and global variables
Component animation	4	4	4	4	4	1
Animation customization	1	1	3	1	1	1

	Speed controls	Interaction controls	Visualization controls
Interaction controls	3*	1	4

*Bpmn2Animator does not support the possibility of navigating directly to a date. For that reason it receives a score of 3 instead of 4.

By analyzing the final score, using the same formula that was used in the tool survey, we see that Bpmn2Animator obtains a higher score than the modeling dedicated tools presented in the tool survey. On the other hand, the plug-in score is lower than Arena and Simul8 due to the animation customization capabilities that it does not have.

Table 59 - Bpmn2Animator score

Tool	Score 4	Score 3	Score 2	Score 1	FINAL SCORE
Bpmn2Animator	6	2	0	7	37

8.2. Future work

It is possible to divide the future work for the developed work in three components:

- Animation capabilities evaluation taxonomy application;
- Animation customization;
- Bpmn2Animator plug-in improvement and new features;

The proposed animation capabilities evaluation taxonomy could be used to evaluate a larger sample of BPM suites and tools and other generic simulation/monitoring tools to get further conclusions about the animation and customization features of the state of the art.

Animation customization features were included in the proposed taxonomy and evaluated for all tools presented in the tools survey chapter. However, customization mechanisms were not explored further in this dissertation. For instance it is worth investigating how to reduce the learning curve and effort required for the animations preparation.

The developed prototype could be enriched with new features and capabilities that were idealized during the development of our work, but fell out of its scope. The ideas that could be explored are the following:

- Connect the Bpmn2Animator to an open-source BPM engine like, for example, the JBPM. This would provide processes execution data in real time to our plug-in, and endow it with monitoring capabilities.
- Another point that could be improved is the metrics that are available in the prototype that can be chosen to animate the Activities, Queues and Performers during animation. New metrics should be studied and implemented in the Bpmn2Animator.
- The animation interactivity controls could also be enhanced. In the present version of the prototype only play, pause, stop and animation speed controls are available. It would be nice for process execution replay to have the ability to choose the exact date and time one wants to visualize.

Our proposed BPMN extension could be an initial effort for the normalization of BPMN animation that could be introduced in a future version of the notation specification. Such normalization work should be done together with the Object Management Group.

Bibliography

- AALST, W. M. P. Business Process Simulation Revisited Enterprise and Organizational Modeling and Simulation. In J. BARJIS ed.: Springer Berlin Heidelberg, 2010, vol. 63, p. 1-14.
- AALST, W. M. P. V. D., M. D. LEONI AND A. H. M. T. HOFSTEDE. Process Mining And Visual Analytics: Breathing life into business process models. In A. FLOARES ed. *Computational Intelligence*. Hauppauge, NY, USA Nova Science Publishers, 2012.
- Activiti. Activiti, 2013 [Viewed 15 January 2013]. Available from: <http://www.activiti.org/>
- AGARWAL, R. 2011. Software development process animation. In *Proceedings of the Proceedings of the 49th Annual Southeast Regional Conference*, Kennesaw, Georgia, 2011. ACM, 2016098, 221-226.
- ALLWEYER, T. *Bpmn 2.0*. 1st ed.: Books on Demand GmbH, 2010. ISBN 9783839149850.
- ALTIOK, T. AND B. MELAMED *Simulation Modeling And Analysis With Arena*. 1st ed.: Academic Press, 2007. ISBN 9780123705235.
- Arena Simulation Software. Automation, Rockwell 2014 [Viewed 07-06-2014]. Available from: http://www.arenasimulation.com/Arena_Home.aspx
- Open Source BPM - Innovative studio for process modeling. Bonitasoft, 2013 [Viewed 14 January 2013]. Available from: <http://www.bonitasoft.com/>
- CHUNG, C. A. *Simulation Modeling Handbook: A Practical Approach*. CRC Press, Inc., 2003. 608 p. ISBN 0849312418.
- CORREIA, A. C. Quality of Process Modeling Using BPMN: A Model-Driven Approach Universidade de Lisboa, 2014.
- CROWSTON, K., H. ANNABI, J. HOWISON AND C. MASANGO Defining Open Source Software Project Success. Proceedings of the 24th International Conference on Information Systems (ICIS), 2003.
- CROWSTON, K., H. ANNABI, J. HOWISON AND C. MASANGO. Towards a portfolio of FLOSS project success measures. In *Collaboration, Conflict and Control: The 4th Workshop on Open Source Software Engineering, International Conference on Software Engineering (ICSE 2004)*. 2004.
- DE VREEDE, G.-J. AND A. VERBRAECK Animating organizational processes Insight eases change. *Simulation Practice and Theory*, 1996, 4(4), 245-263.
- DENNIS, S., B. KING, M. HIND AND S. ROBINSON 2000. Applications of business process simulation and lean techniques in British Telecommunications PLC. In *Proceedings of the 32nd*

conference on Winter simulation, Orlando, Florida, 2000. Society for Computer Simulation International, 510673, 2015-2021.

Programming Language Comparison. 2014 [Viewed 13-05-2014]. Available from: <http://programming.dojo.net.nz/resources/programming-language-comparison/index>

DU, X., C. GU AND N. ZHU 2012. A survey of business process simulation visualization. In *Proceedings of the International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering (ICQR2MSE)*, 2012. IEEE, 43-48.

FERREIRA, K. A. M., M. A. S. BIGONHA, R. S. BIGONHA, L. F. O. MENDES, et al. Identifying thresholds for object-oriented software metrics. *Journal of Systems and Software*, 2012.

FERREIRA, P. A. D. S. Implementação de processos da fase de operação de serviço do ITIL® em ambiente universitário - O caso do ISCTE-IUL. ISCTE-IUL, 2011.

FOWLER, M. AND K. BECK *Refactoring: Improving the Design of Existing Code*. 1st ed.: Addison-Wesley Professional, 1999. ISBN 0201485672.

GREASLEY, A. 2000. Process and enterprise improvement: effective uses of business process simulation. In *Proceedings of the 32nd conference on Winter simulation*, 2000. Society for Computer Simulation International, 2004-2009.

HAMMER, M. What is Business Process Management? *Handbook on Business Process Management*. In J.V. BROCKE AND M. ROSEMANN eds.: Springer Berlin Heidelberg, 2010, p. 3-16.

HARRINGTON, H. J. *Business process improvement: the breakthrough strategy for total quality, productivity, and competitiveness*. McGraw-Hill, 1991. ISBN 9780070267688.

HEVNER, A. R., S. T. MARCH, J. PARK AND S. RAM Design Science in Information Systems Research. *MIS Quarterly*, 2004, 28, 75-105.

HLUPIC, V. AND S. ROBINSON 1998. Business process modelling and analysis using discrete-event simulation. In *Proceedings of the 30th conference on Winter simulation*, Washington, D.C., United States, 1998. IEEE Computer Society Press, 293489, 1363-1370.

WebSphere Business Modeler Advanced. IBM, 2012 [Viewed 28 Oct 2012]. Available from: <http://www-01.ibm.com/software/integration/wbimodeler/advanced/>

INDIHAR-STEMBERGER, M., A. POPOVIC AND V. BOSILJ-VUKSIC 2003. Simulation and Information systems modelling: A framework for business process change. In *Proceedings of the 15th European Simulation Symposium*, 2003.

JANSSEN, M., A. JOHA AND A. ZUURMOND Simulation and animation for adopting shared services: Evaluating and comparing alternative arrangements. *Government Information Quarterly*, 2009, 26(1), 15-24.

jBPM - JBoss Community. 15 January 2013]. Available from: <http://www.jboss.org/jbpm>

KELTON, W. D., R. P. SADOWSKI AND D. A. SADOWSKI *Simulation with Arena*. McGraw-Hill, 2002. ISBN 9780072392708.

LANNING, D. L. AND T. M. KHOSHGOFTAAR. Canonical Modeling of Software Complexity and Fault Correction Activity. In., 1994, p. 374-381.

MARTIN, R. C. *Agile Software Development: Principles, Patterns, and Practices*. 1st ed.: Prentice Hall, 2003. ISBN 0135974445.

MARTIN, R. C. *Clean code: a handbook of agile software craftsmanship*. Prentice Hall PTR, 2009.

MCCABE, T. J. A complexity measure. In *Proceedings of the 2nd international conference on Software engineering*. 1976.

MILLER, G. The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information. In *The Psychological Review*, 1956.

Modelio Open Source Community. ModelioSoft, 14 January 2013]. Available from: <http://www.modelio.org/>

OMG *Business Process Model and Notation (BPMN) version 2.0*. 2011.

PEGDEN, C. D., R. E. SHANNON AND R. P. SADOWSKI *Introduction to simulation using SIMAN*. McGraw-Hill, 1995. ISBN 9780070493209.

PRESSMAN, R. S. *Software engineering: a practitioner's approach*. McGraw-Hill Higher Education, 2010. ISBN 9780073375977.

PROGRESS *Progress Savvion Business Manager 8.0: Process Modeler User's Guide*. Progress, 2012.

REKAPALLI, P. V. AND J. C. MARTINEZ 2007. A message-based architecture to enable runtime user interaction on concurrent simulation-animations of construction operations. In *Proceedings of the 39th conference on Winter simulation: 40 years! The best is yet to come*, Washington D.C., 2007. IEEE Press, 1351902, 2028-2031.

ROHRER, M. W. 2000. Seeing is believing: the importance of visualization in manufacturing simulation. In *Proceedings of the 32nd conference on Winter simulation*, Orlando, Florida, 2000. Society for Computer Simulation International, 510552, 1211-1216.

- SARGENT, R. G. 1998. Verification and validation of simulation models. In *Proceedings of the 30th conference on Winter simulation*, Washington, D.C., United States, 1998. IEEE Computer Society Press, 293216, 121-130.
- SHÖN, D. A. The Reflective Practitioner: How Professionals Think in Action. In. New York: Basic Books, 1983.
- SILVER, B. *BPMN Method and Style: A Levels-based Methodology for BPM Process Modeling and Improvement Using BPMN 2. 0*. Cody-Cassidy Press, 2009. ISBN 9780982368107.
- SIMON, H. A. The Sciences of the Artificial. In. Cambridge: MIT Press, 1996.
- SINUR, J. AND J. B. HILL. Magic Quadrant for Business Process Management Suites. 2010.
- STEINBERG, D., F. BUDINSKY, M. PATERNOSTRO AND E. MERKS *EMF: Eclipse Modeling Framework*. 2009.
- TAYLOR, S., D. CANNON AND D. WHEELDON *ITIL V3 Service Operation*. The Office of Government Commerce, 2011.
- TERRENCE, P. AND L. KAPILA IDEF based methodology for rapid data collection. *Integrated Manufacturing Systems*, 2001, 12(3), 187-194.
- TEWOLDEBERHAN, T. W., VERBRAECK, A., VALENTIN, E., BARDONNET, G. 2002. An evaluation and selection methodology for discrete-event simulation software. In *Proceedings of the Winter Simulation Conference*, 2002. 67 - 75.
- WHITE, S. A. AND D. MIERS *Bpmn Modeling and Reference Guide*. Future Strategies Inc, 2008. ISBN 9780977752720.
- WYNN, M. T., M. DUMAS, C. J. FIDGE, A. H. M. T. HOFSTEDE, et al. 2008. Business process simulation for operational decision support. In *Proceedings of the International conference on Business process management*, Brisbane, Australia, 2008. Springer-Verlag, 1793723, 66-77.
- Yaoqiang BPMN Editor. Yaoqiang, 2013 [Viewed 14 January 2013]. Available from: <http://sourceforge.net/projects/bpmn/?source=dlp>

Annexes

A. OSS Projects web addresses

Table 60 presents the web addresses for the homepage, source code repository and forum of each tool that was analyzed in chapter 0.

Table 60 - OSS projects web addresses

OSS Projects web addresses	
Activiti	
Homepage	http://www.activiti.org/
Repository	https://github.com/Activiti/Activiti (https://github.com/Activiti/Activiti-Designer)
Forum	http://forums.activiti.org/forums/activiti-users
Bonita Open Solution: Open Source BPM	
Homepage	http://www.bonitasoft.com/
Repository	http://sourceforge.net/projects/bonita/?source=directory
Source Code Repository	http://svn.bonitasoft.org/
Forum	http://www.bonitasoft.org/forum/
BPMN2 Modeler	
Homepage	http://www.eclipse.org/bpmn2-modeler/
Repository	https://github.com/eclipse/bpmn2-modeler
Forum	http://www.eclipse.org/forums/index.php
jBPM	
Homepage	http://www.jboss.org/jbpm
Repository	https://github.com/droolsjbpm/jbpm-designer/commits/master
Forum	https://community.jboss.org/en/jbpm/dev?view=discussions
Modelio – Modeling environment (UML)	
Homepage	http://www.modelio.org/index.php
Repository	http://sourceforge.net/projects/modeliouml/?source=directory
Forum	http://www.modelio.org/forum/index.html
Yaoqiang BPMN Editor	
Homepage	http://bpmn.sourceforge.net/
Repository	http://sourceforge.net/projects/bpmn/?source=directory
Forum	http://sourceforge.net/p/bpmn/discussion/1152583/

B. Bpmn2Animator development

In this section, the annexes related with the Bpmn2Animator development are presented.

B.1 Bpmn2Animator extension elements of Bpmn2 Modeler

Table 61 presents a description for each extension element of the Bpmn2Animator. Several Bpmn2 Modeler extension point are used by our prototype in order to contribute with new functionalities to the modeler plug-in.

Table 61 – Bpmn2Animator extension elements

Type	Id	Description
runtime	org.quasar.bpmn2.animator.runtime	Defines the runtime extension properties like the name, id and class.
Model	Bpmn2AnimatorModelResourceFactoryImpl	Defines the Model Factory implementation to be used in the runtime extension.
featureContainer	TaskCustomFeatureContainer	This <i>featureContainer</i> is used to define the <i>customFeature</i> that is used to animate the Task element.
featureContainer	GatewayCustomFeatureContainer	This <i>featureContainer</i> is used to define the <i>customFeature</i> that is used to animate the Gateway element.
featureContainer	SequenceCustomFeatureContainer	This <i>featureContainer</i> is used to define the <i>customFeature</i> that is used to animate the <i>SequenceFlow</i> element.
featureContainer	EventCustomFeatureContainer	This <i>featureContainer</i> is used to define the <i>customFeature</i> that is used to animate the <i>Event</i> element.
customTask	Instance	Defines the custom task <i>Instance</i> that is used to present the process instances.
customTask	Queue	Defines the custom task <i>Queue</i> that is used to present the tasks queues in the process model.
customTask	HumanPerformer	Defines the custom task <i>HumanPerformer</i> that is used to present the tasks <i>ResourceRoles</i> that use human resources.
customTask	AutomaticPerformer	Defines the custom task <i>AutomaticPerformer</i> that is used to present the tasks <i>ResourceRoles</i> that use automatic resources.

modelExtension	SequenceFlow Extension	Defines the attributes that are added to the <i>extensionValues</i> property of <i>SequenceFlow</i> .
modelExtension	Performer Extensions	Defines the attributes from the model that are added to the <i>extensionValues</i> property of <i>HumanPerformer</i> and <i>AutomaticPerformer</i> .
modelExtension	Task Extensions	Defines the attributes from the model that are added to the <i>extensionValues</i> property of <i>Task</i> .
modelExtension	UserTask Extensions	Defines the attributes from the model that are added to the <i>extensionValues</i> property of <i>UserTask</i> .
modelExtension	ManualTask Extensions	Defines the attributes from the model that are added to the <i>extensionValues</i> property of <i>ManualTask</i> .
propertyTab	Performer Animation	Defines the class which is responsible for the <i>Performer</i> animation <i>propertyTab</i> which is used in the <i>Performer</i> elements (<i>CustomTask</i>). This property tab is used to define the animation configuration parameters.
propertyTab	Task Animation	Defines the class which is responsible for the <i>Task</i> animation <i>propertyTab</i> which is used in all task elements. This property tab is used to define the animation configuration parameters.
propertyTab	Task Animation Attributes	This <i>propertyTab</i> is used to present the <i>Task</i> attributes that are calculated during the animation of the process.
propertyTab	TextAnnotationTabAttributes	This <i>propertyTab</i> is used to present the <i>Performer</i> and <i>Queue</i> attributes that are calculated during the animation of the process.
propertyTab	SequenceFlowTabAttributes	This <i>propertyTab</i> is used to present the <i>SequenceFlow</i> attributes that are calculated during the animation of the process.
propertyTab	BoundaryEventTabAttributes	This <i>propertyTab</i> is used to present the <i>BoundaryEvent</i> attributes that are calculated during the animation of the

		process.
propertyTab	GatewayTabAttributes	This <i>propertyTab</i> is used to present the <i>Gateway</i> attributes that are calculated during the animation of the process.
modelEnablement	Default	Defines which BPMN2 elements and <i>customTasks</i> should be enabled in the runtime extension when modelling a Process from the type "Process".
modelEnablement	Default Collaboration	Defines which BPMN2 elements and <i>customTasks</i> should be enabled in the runtime extension when modelling a "Collaboration" process type.

C. Service request fulfillment process data preparation

In this section are presented the annexes related to the data preparation of the collected data from the IT service management system. First, the internals of the CSV Analyzer plug-in that was developed to facilitate the data preparation. Also in the following subsections are presented the data collected from the IT service management system that is related with the resources groups, and actions types and descriptions.

C.1 CSV Analyzer internals

CSV Analyzer is an Eclipse IDE plug-in that we developed in order to facilitate the data preparation, in order to make the collected data from the IT service management system ready to be consumed by Bpmn2Animator. Herein, we present the internals of the CSV Analyzer internals.

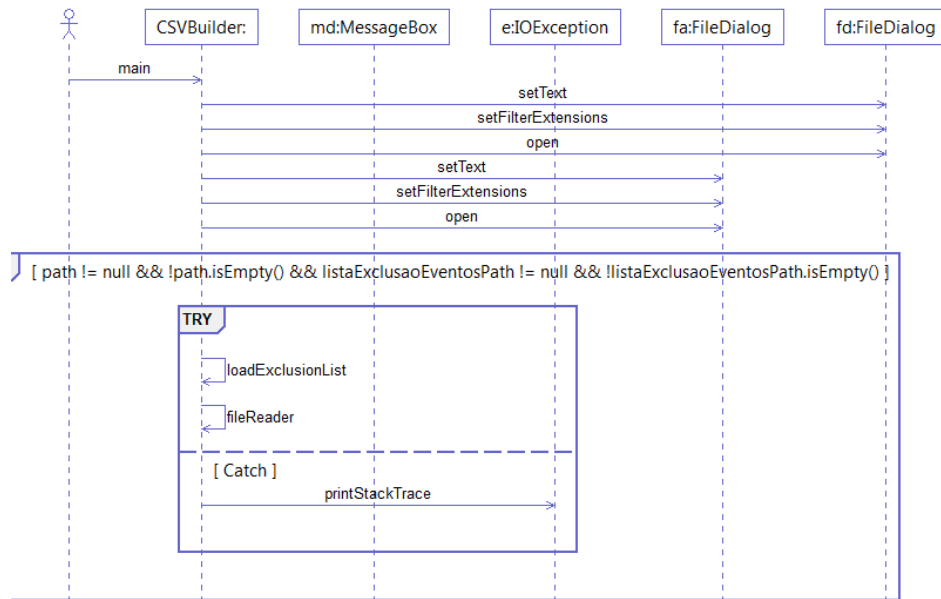


Figure 99 - CSV Analyzer main method UML sequence diagram

In Figure 99 is presented the UML sequence diagram for the CSV Analyzer main method. Besides the commands to show the *FileDialogs* to the user, this method executes the *loadExclusionList* method and the *fileReader* method, which contains the cycle responsible for inspecting all events in the chosen CSV file. The *loadExclusionList* method loads the list of events that will be ignored and, therefore, will not appear in the processed events file. The *fileReader* method reads the events from the CSV file one by one. As it reads the events it evaluates if the event should be ignored by checking if the event type exists in the exclusion list (see Table 65). When all events from one instance have been read and inserted on a temporary list, the method executes the *applyRulesToEvents* method, which UML sequence diagram is presented in Figure 100.

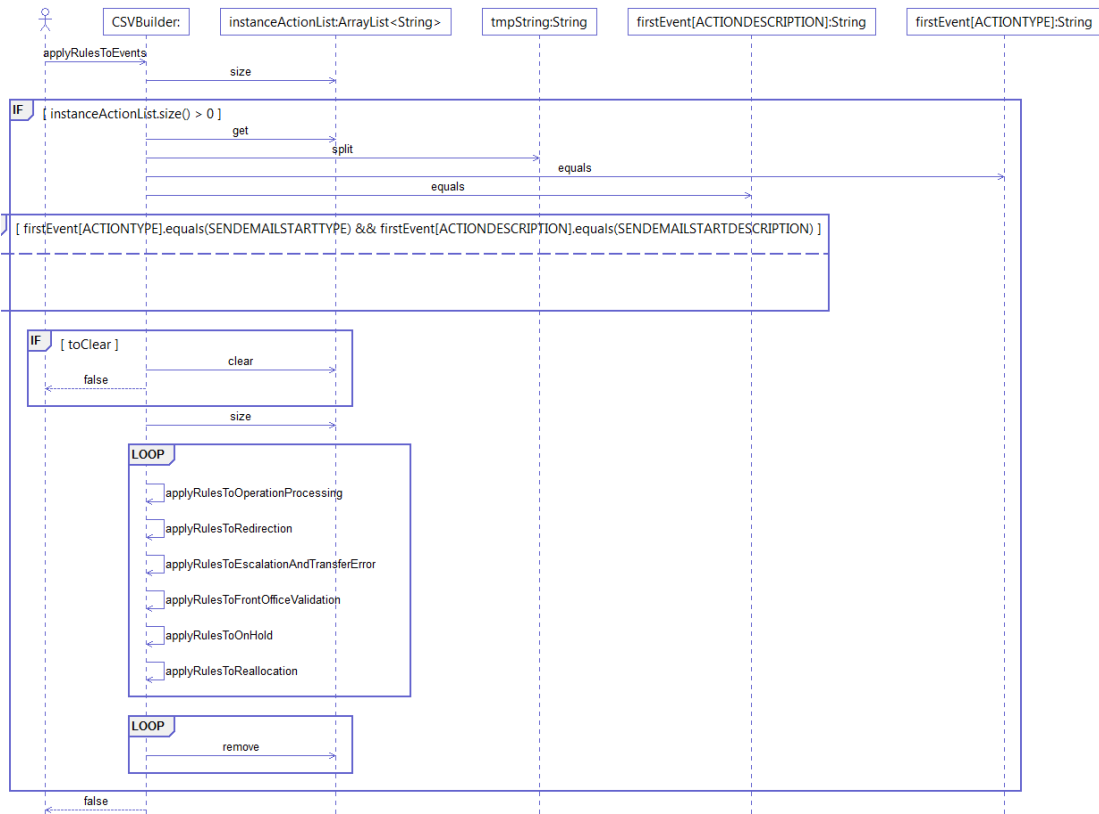


Figure 100 - applyRulesToEvents UML sequence diagram

Basically, what the applyRulesToEvents does is to execute the several existent methods that apply the rules for the instance events. Each method is explained in Table 62.

Table 62 - Applying data preparation rules methods description

Method	Description	
applyRulesToOperationProcessing	Problems	1 - Identify the group to where the next activity will be sent in order to use this information to decide the path to travel in the gateways decisions. 2 - Identify the next activity type in order to use this information to decide the path to travel in the gateways decisions.
	Solution	1 - Searches the next action of "OperationProcessing" type and sets its "group_level" value to the "escalation_group" column of the action being analyzed. 2 - Searches the next action of "OperationProcessing" type and sets its "Description" value to the "next_activity" column of the action being analyzed.

applyRulesToRedirection	Problems	1 - "Redirection" actions are not interpreted by the Bpmn2Animator event generator so they have to be removed from the processed events. However, these actions appear always between two "Operation Processing" type actions and if only the "Redirection" action is removed, then two repeated "Operation Processing" actions will remain.
	Solution	1 - Remove the "Redirection" actions together with the predecessor "Operation Processing" action.
applyRulesToEscalationAndTransferError	Problems	1 - "Escalation" and "Transfer Error" appear always between two "Operation Processing" actions. Because each action of "Operation Processing" in the Bpmn2Animator represents an activity execution this leads to duplication of activities execution and therefore inconsistency and errors during execution of the events in the plug-in. 2 - Identify the group to whom the activity was escalated, in the case of an "Escalation" action, or returned to, in the case of a "Transfer Error" action, in order to this information to decide the path to travel in the gateways decisions. 3 - Sometimes the "Transfer Error" returns an activity to the same group what leads to execution problems.
	Solution	1 - Remove the predecessor "Operation Processing" action of each "Escalation" and "Transfer Error" action. 2 - Search the next action of "OperationProcessing" type and sets its "group_level" value to the "escalation_group" column of the action being analyzed. 3 - Evaluate if the "Transfer Error" returns the activity to the same group and if it is true, then remove the "Transfer Error" action from the list.
applyRulesToFrontOfficeValidation	Problems	1 - It is necessary to understand if the instance is completed in the "Front Office Validation with Notation" action.
	Solution	1 - Evaluates if the "Front Office Validation with Notation" is the last action for the instance and if so fill the "is_Finished" column with the value "1".

applyRulesToOnHold	Problems	1 - The "On Hold" and "End of On-Hold" actions have to be removed, but these actions should be taken into account in order to calculate accurately the waiting times of the activities queues. Notice that when there is an "On hold" and "End of On-Hold" the waiting time should not be counted.
	Solution	1 - The actions "On Hold" and "End of On-Hold" are removed. However, before being removed, the next "Operation Processing" action "On_hold" column is filled with the "1" value which will be read later by the Bpmn2Animator plug-in and it will understand that the waiting time should not be taken into account.
applyRulesToReallocation	Problems	1 - It is necessary to remove instances that are clearly instances from the Incident Management process. 2 - The "Reallocation" actions are not understood by the Bpmn2Animator plug-in and therefore they should be ignored.
	Solution	1 - Evaluates if the instance topic refers to an incident instead of referring to a Service Request Fulfilment instance. 2 - Evaluates if the action is of the type "Reallocation" and, if so, removes it from the list.

C.2 Sort helper list

Some actions collected from the IT service management related to the same service request, have the same "real end date", which is the date when they were executed. In order to process the actions in the CSV Analyzer plug-in, the collected data should be sorted correctly, and in order to do so, we created the sort level that we used to sort the collected actions. Table 63 presents the sort level for each action type.

Table 63 - SortHelper content

Type	Sort level
End of On-Hold,End of On-Hold	7
Escalation,Análise e Classificação	4
Escalation,Fornecer serviço	6
Front Office Validation with Notation,Confirmação e avaliação do fornecimento do serviço	9
Front Office Validation with Notation,Notificação de serviço fornecido.	9
On Hold,On Hold	6
Operation Processing,Análise e Classificação	3
Operation Processing,Fornecer serviço	5
Reallocation,Reallocation	2
Redirection,Análise e Classificação	4
Redirection,Fornecer serviço	6
Send Email,User notification	1
Transfer Error,Análise e Classificação	4
Transfer Error,Fornecer serviço	6

C.3 IT service management system actions description

Table 64 presents the system action types and a brief description for each one.

Table 64 - IT service management system actions descriptions

Action Type	Description
Redirection	Indicates that the service request was passed from one person to another from the same team.
Operation Processing	Indicates a human activity accomplished by somebody from the IT Department team. An Operation Processing can be of the type "Analyze and classify" or "Supply service".
Escalation	Indicates that an activity has been escalated to a superior group.
Transfer Error	Indicates that a previous escalation or Reallocation has been inappropriate and the service request will return back to the team that asked for the escalation.
Immediate solution	It is an incident management process activity. This activity is realized when the incident is immediately solved.
Reallocation	Indicates a change in the topic of the Incident or Service Request which is selected from the catalog. This action can represent a change of the instance from an Incident to a Service Request or the other way around.

C.4 Actions to ignore list content

We were not able to export only the data for the service request process from the IT service management system, which means that actions from other processes were collected also. In order to ignore those actions, we listed all of them and used that list when we executed the CSV Analyzer plug-in in the data preparation phase. The list of the actions that should be ignored is presented in Table 65.

Table 65 - Actions to ignore

Type	Description
Send Email	Notification to requesting Person
Send Email	Send Email
Send Email	Send Email to Contact
Send Email	Information for Support Person
Send Email	Notificação aprovação serviço ao utilizador
Send Email	Send Email to Support Persons
Assign	Assignment
Outgoing user call	Outgoing user call
Incoming email	Incoming email
Registo de MAC	Registo de MAC
Outgoing level 2 call	Outgoing level 2 call
Self Service Request	Front Office
New RFC	Creation
Videoconferencing	Videoconferencing
Updated Urgency	Urgency change
Cancellation by Manager or Recipient	Cancellation by Manager or Recipient
Incoming user call	Incoming user call
Updated by user	Updated by User
Operation Approval with Authentication	Autorização do Pedido
Escalation	Autorização do Pedido
Immediate solution	Immediate solution
Operation Processing	Análise e Resolução

C.5 Service request process groups

Table 66 presents the list of groups that are defined in the IT service management system. The group level is equivalent to the lanes on the BPMN process model. A mapping of the group level with the BPMN process model lanes can be seen in Table 67.

Table 66 - Groups defined in the IT service management system

Group Level	Group
2	1st line Support
3	Comissão Consultiva de Alterações
3	Comissão Consultiva de Alterações de Emergência
3	Desenvolvimento
3	Equipa de Gestão de Alterações
3	Equipa de Gestão de Problemas
2	Gestão de software
3	Gestor de Alteração
3	Gestor de Problema
1	Helpdesk
3	Infraesturas (Storage, VMWARE, Monitorização)
2	IT Labs support
3	Redes
3	SI
3	Sistemas
2	Suporte a Auditórios
2	Suporte a Biblioteca e Sala de Estudo
2	Suporte a Docentes
2	Suporte a Funcionários
2	Suporte a impressoras
2	Suporte a listas de correio electrónico
3	Suporte a páginas Web
2	Suporte a salas de aula
2	Suporte a Tecnologias Educativas
2	Suporte a videoconferência
2	Suporte audiovisual
4	Supplier-Fénix
4	Supplier-Gestão cartões
4	Supplier-Portal

D. Service request process model parameterization

This section presents and describes the necessary parameterizations that were made to the process model in order to make the execution of the collected and prepared process execution data. The parameterization is made in the gateways presented in the process model, which are used to make the instances travel the right workflow paths. They are highlighted in Figure 101 with a numbered orange square. This parameterization was useful because the Bpmn2Animator event generator can read the gateway information and match it with the information from the execution data.

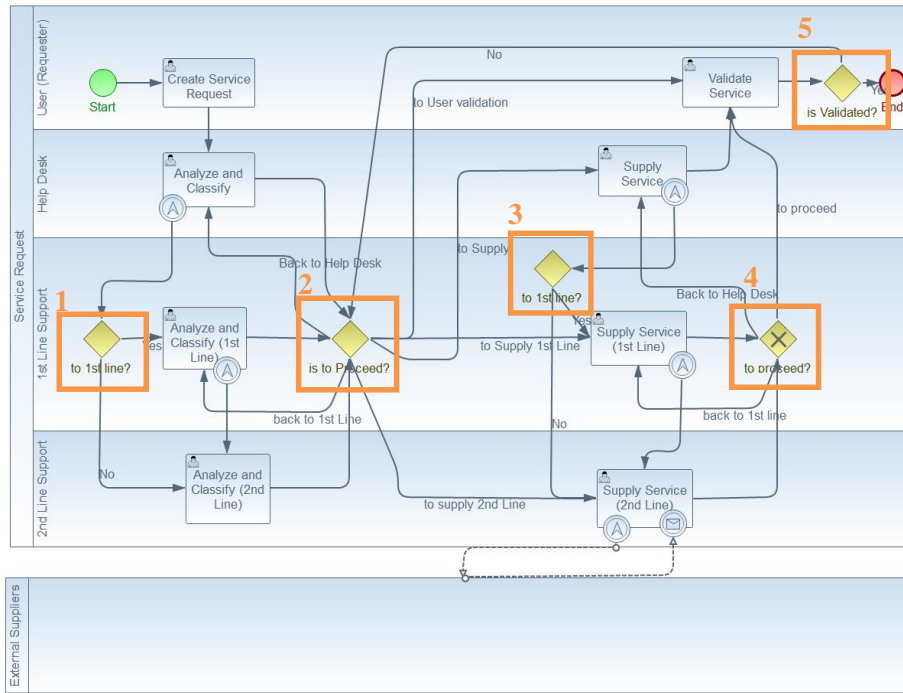


Figure 101 - Service request process model

Gateway 1 (“to 1st line?”) parameterization

The numbers that appear in the conditions of Table 68 represent the group level where the instance should go to. The group level is the equivalent to the lanes in the process model and the mapping between the two can be seen in Table 67. If the “Escalation_Group” column of an action is filled with the value 2, meaning that the next action will be realized by the level 2 group, then the condition for the “Yes” sequenceflow is true and the instance will travel to the “Analyze and classify (1st line)” activity. Otherwise, the instance will travel to the “Analyze and classify (2nd line)” activity.

Table 67 – Group levels mapping with process model lanes

Group level	Process model lane
1	Help desk
2	1st line support
3	2nd line support
4	External suppliers

Table 68 - Gateway 1 (“to 1st line?”) conditions

SequenceFlow	Condition
Yes	2
No	3

Gateway 2 (“is to proceed?”) parameterization

This gateway is used to evaluate several conditions. Basically it is used to determine every possible paths for the following activities: “Analyze and classify”, “Analyze and classify (1st line)” and “Analyze and classify (2nd line)”.

The first condition, “PROCEED”, is evaluated as true if the column “next_activity” for the action is empty and if the action is not a "Transfer Error" type action.

The “TRANSFERERROR==1” and the “TRANSFERERROR==2” conditions evaluate true if the action is of the type "Transfer Error" and also, in first case if the “Escalation_Group” is equal to 1 or in second case if it is equal to 2. If the first condition is true, the instance will travel through the “back to help desk” SequenceFlow to the “Analyze and classify” activity. On the other hand, if the second condition is true, the instance will travel through the “back to 1st line” SequenceFlow to the “Analyze and classify 1st line”.

Finally, the other three conditions, “NEXTACTIVITY=="Fornecer Serviço" AND GROUP==2”, “NEXTACTIVITY=="Fornecer Serviço" AND GROUP==1” and “NEXTACTIVITY=="Fornecer Serviço" AND GROUP==3” are similar. First, the NEXTACTIVITY=="Fornecer Serviço” evaluates if the column “next_activity” is filled with the value “Fornecer Serviço” which means that the process instance should be travelling to one of the “supply service” activities. The “GROUP==1” part of the condition indicates to which of the “supply service” activities the process instance will travel to. If the column “Escalation_Group” is filled with the value “1”, then the “NEXTACTIVITY=="Fornecer Serviço" AND GROUP==1” condition will be true and the process instance will travel through the “to supply” SequenceFlow to the “Supply Service” activity. If the “Escalation_Group” is filled with the value “2”, the process instance will travel to the “Supply Service (1st line)” and if the “Escalation_Group” is filled with the value “3”, then the instance will travel to the “Supply Service (2nd line)”.

Table 69 - Gateway 2 (“is to proceed?”) conditions

SequenceFlow	Condition
to user validation	PROCEED
back to help desk	TRANSFERERROR==1
back to 1 st line	TRANSFERERROR==2
to supply 1 st line	NEXTACTIVITY=="Fornecer Serviço" AND GROUP==2
to supply	NEXTACTIVITY=="Fornecer Serviço" AND GROUP==1
to supply 2 nd line	NEXTACTIVITY=="Fornecer Serviço" AND GROUP==3

Gateway 3 (“to 1st line?”) parameterization

This condition is similar in all aspects to the one described in the point 1.

Table 70 - Gateway 3 (“to 1st line?”) conditions

SequenceFlow	Condition
Yes	2
No	3

Gateway 4 (“to proceed?”) parameterization

The three conditions presented in Table 71 are similar to the first three conditions presented in the point 2 of this section, with difference that instead of travelling to “Analyze and classify” activities the process instances will travel to “Supply Service” activities and in the last case they will travel to the “Validate Service” activity.

Table 71 - Gateway 4 (“to proceed?”) conditions

SequenceFlow	Condition
back to 1st line	TRANSFERERROR==2
back to help desk	TRANSFERERROR==1
to proceed	PROCEED

Gateway 5 (“is validated?”) parameterization

This gateway is used to evaluate if a determined action ends the process. If the column “is_Finished” for an “Operation Processing” action is filled with the value “1” then the “ISFINISHED” condition will be evaluated as true and the instance will travel through the “Yes” SequenceFlow. Otherwise, if the same column is filled with value “0” then the “REJECTED” condition will be evaluated as true and the instance will travel through the “No” SequenceFlow.

Table 72 - Gateway 5 (“is validated?”) conditions

SequenceFlow	Condition
Yes	ISFINISHED
No	REJECTED

E. Service request fulfillment process replay results

This section presents the replay results gathered from the Bpmn2Animator when replaying the service request fulfillment process data execution.

E.1 11th of September of 2013

Table 73 - Queues results

Activities	Number of processed instances	Number of SLAS's violations
Create Service Request	105	0
Validate Service	5	0
Analyze And Classify	90	18
Supply Service	40	0
Analyze and Classify (1st line)	0	0
Supply Service (1st line)	19	1
Analyze and Classify (2nd line)	2	0
Supply Service (2nd line)	18	0

Table 74 - Performers results

Queues	Total instances	Total queue waiting time (minutes)	Queue average wait time (minutes)	Instances waiting in the queue	Queue maximum size
Create Service Request	105	0	0	0	1
Validate Service	74	113	1	69	69
Analyze And Classify	105	62989	599	15	22
Supply Service	41	9576	233	1	3
Analyze and Classify (1st line)	0	0	0	0	0
Supply Service (1st line)	27	33677	1347	8	9
Analyze and Classify (2nd line)	2	1650	825	0	2
Supply Service (2nd line)	25	6930	277	7	9

Performers	Total processed instances	Total of working minutes
User (Create Service Request)	105	0
User (Validate Service)	5	0
Help Desk (Analyze And Classify)	90	900
Help Desk (Supply Service)	40	400
1st line (Analyze And Classify)	0	0
1st line (Supply Service)	19	190
2nd line (Analyze And Classify)	2	20
2nd line (Supply Service)	18	180

Table 75 - Gateways results Table 76 - Boundary events results

Table 77 - Gateways results

Gateways	Number of processed instances
to 1st line? (analyze and classify)	2
is to Proceed? (analyze and classify)	90
to proceed? (supply service)	31
to 1st line? (supply service)	8

Table 78 - Boundary events results

Boundary Event	Number of processed instances
Analyze And Classify	2
Supply Service	8
Analyze and Classify (1st line)	0
Supply Service (1st line)	6

Table 79 - Sequence flows results

Sequence Flows	Number of processed instances
Start --> Create Service Request	105
Create Service Request --> Analyze And Classify	105
Analyze And Classify --> is to proceed?	88
Analyze And Classify --> to 1st line?	2
to 1st line? --> Analyze and Classify (1st line)	0
to 1st line? --> Analyze and Classify (2nd line)	2
Analyze and Classify (1st line) --> is to proceed?	0
Analyze and Classify (1st line) --> Analyze and Classify (2nd line)	0
Analyze and Classify (2nd line) --> is to proceed?	2
(from "is to proceed?") Back to Help Desk	0
(from "is to proceed?") Back to 1st line	0
(from "is to proceed?") to user Validation	16
(from "is to proceed?") to Supply	39
(from "is to proceed?") to Supply 1st line	24
(from "is to proceed?") to Supply 2nd line	11
Supply Service --> Validate Service	32
Supply Service --> to 1st line?	8
to 1st line? --> Supply Service (1st line)	0
to 1st line? --> Supply Service (2nd line)	8
Supply Service (1st line) --> to proceed?	13
Supply Service (1st line) --> Supply Service (2nd line)	6
Supply Service (2nd line) --> to proceed?	18
(from "to proceed?") Back to 1st line	3
(from "to proceed?") Back to Help Desk	2
(from "to proceed?") to proceed	26
validate service --> is validated?	5
is validated? --> End	5
is validated? --> is to proceed?	0

E.2 19th of September of 2013

Table 80 - Activities results

Activities	Number of processed instances	Number of SLAS's violations
Create Service Request	213	0
Validate Service	38	0
Analyze And Classify	199	51
Supply Service	84	0
Analyze and Classify (1st line)	0	0
Supply Service (1st line)	54	7
Analyze and Classify (2nd line)	2	0
Supply Service (2nd line)	37	0

Table 81 - Queues result

Queues	Total instances	Total queue waiting time (minutes)	Queue average wait time (minutes)	Instances waiting in the queue	Queue maximum size
Create Service Request	213	0	0	0	1
Validate Service	180	86567	483	142	160
Analyze And Classify	213	220253	1034	14	22
Supply Service	84	18195	216	0	4
Analyze and Classify (1st line)	0	0	0	0	0
Supply Service (1st line)	67	129127	1986	13	21
Analyze and Classify (2nd line)	2	1650	825	0	2
Supply Service (2nd line)	47	46974	999	10	14

Table 82 - Performers results

Performers	Total processed instances	Total of working minutes
User (Create Service Request)	213	0
User (Validate Service)	38	0
Help Desk (Analyze And Classify)	199	1990
Help Desk (Supply Service)	84	840
1st line (Analyze And Classify)	0	0
1st line (Supply Service)	54	540
2nd line (Analyze And Classify)	2	20
2nd line (Supply Service)	37	370

Gateways	Number of processed instances
to 1st line? (analyze and classify)	2
is to Proceed? (analyze and classify)	203
to proceed? (supply service)	81
to 1st line? (supply service)	13

Table 83 - Gateways results

Boundary Event	Number of processed instances
Analyze And Classify	2
Supply Service	13
Analyze and Classify (1st line)	0
Supply Service (1st line)	10

Table 84 - Boundary events results

Table 85 - Sequence flows results

Sequence Flows	Number of processed instances
Start --> Create Service Request	213
Create Service Request --> Analyze And Classify	213
Analyze And Classify --> is to proceed?	197
Analyze And Classify --> to 1st line?	2
to 1st line? --> Analyze and Classify (1st line)	0
to 1st line? --> Analyze and Classify (2nd line)	2
Analyze and Classify (1st line) --> is to proceed?	0
Analyze and Classify (1st line) --> Analyze and Classify (2nd line)	0
Analyze and Classify (2nd line) --> is to proceed?	2
(from "is to proceed?") Back to Help Desk	0
(from "is to proceed?") Back to 1st line	0
(from "is to proceed?") to user Validation	38
(from "is to proceed?") to Supply	79
(from "is to proceed?") to Supply 1st line	61
(from "is to proceed?") to Supply 2nd line	25
Supply Service --> Validate Service	71
Supply Service --> to 1st line?	13
to 1st line? --> Supply Service (1st line)	1
to 1st line? --> Supply Service (2nd line)	12
Supply Service (1st line) --> to proceed?	44
Supply Service (1st line) --> Supply Service (2nd line)	10
Supply Service (2nd line) --> to proceed?	37
(from "to proceed?") Back to 1st line	5
(from "to proceed?") Back to Help Desk	5
(from "to proceed?") to proceed	71
validate service --> is validated?	38
is validated? --> End	34
is validated? --> is to proceed?	4

E.3 26th of September of 2013

Table 86 - Activities results

Activities	Number of processed instances	Number of SLAS's violations
Create Service Request	302	0
Validate Service	219	0
Analyze And Classify	298	84
Supply Service	126	0
Analyze and Classify (1st line)	0	0
Supply Service (1st line)	99	18
Analyze and Classify (2nd line)	3	0
Supply Service (2nd line)	52	3

Table 87 - Queues results

Queues	Total instances	Total queue waiting time (minutes)	Queue average wait time (minutes)	Instances waiting in the queue	Queue maximum size
Create Service Request	302	0	0	0	1
Validate Service	292	2731138	9824	73	192
Analyze And Classify	302	324748	1078	4	27
Supply Service	126	51468	408	0	6
Analyze and Classify (1st line)	0	0	0	0	0
Supply Service (1st line)	104	273708	2631	5	21
Analyze and Classify (2nd line)	3	1778	592	0	2
Supply Service (2nd line)	59	124476	2222	7	14

Performers	Total processed instances	total of working minutes
User (Create Service Request)	302	0
User (Validate Service)	219	0
Help Desk (Analyze And Classify)	298	2980
Help Desk (Supply Service)	126	1260
1st line (Analyze And Classify)	0	0
1st line (Supply Service)	99	990
2nd line (Analyze And Classify)	3	30
2nd line (Supply Service)	52	520

Table 88 - Performers results

Gateways	Number of processed instances
to 1st line? (analyze and classify)	3
is to Proceed? (analyze and classify)	304
to proceed? (supply service)	137
to 1st line? (supply service)	17

Table 89 - Gateways results

Boundary Event	Number of processed instances
Analyze And Classify	3
Supply Service	17
Analyze and Classify (1st line)	0
Supply Service (1st line)	14

Table 90 - Boundary events results

Table 91 - Sequence Flows results

Sequence Flows	Number of processed instances
Start --> Create Service Request	302
Create Service Request --> Analyze And Classify	302
Analyze And Classify --> is to proceed?	295
Analyze And Classify --> to 1st line?	3
to 1st line? --> Analyze and Classify (1st line)	0
to 1st line? --> Analyze and Classify (2nd line)	3
Analyze and Classify (1st line) --> is to proceed?	0
Analyze and Classify (1st line) --> Analyze and Classify (2nd line)	0
Analyze and Classify (2nd line) --> is to proceed?	3
(from "is to proceed?") Back to Help Desk	0
(from "is to proceed?") Back to 1st line	0
(from "is to proceed?") to user Validation	57
(from "is to proceed?") to Supply	121
(from "is to proceed?") to Supply 1st line	97
(from "is to proceed?") to Supply 2nd line	29
Supply Service --> Validate Service	109
Supply Service --> to 1st line?	17
to 1st line? --> Supply Service (1st line)	1
to 1st line? --> Supply Service (2nd line)	16
Supply Service (1st line) --> to proceed?	85
Supply Service (1st line) --> Supply Service (2nd line)	14
Supply Service (2nd line) --> to proceed?	52
(from "to proceed?") Back to 1st line	6
(from "to proceed?") Back to Help Desk	5
(from "to proceed?") to proceed	126
validate service --> is validated?	219
is validated? --> End	213
is validated? --> is to proceed?	6