Software Requirements Document
Team 2.2

Table of Contents

Introduction

Purpose

The goal is to create two add-ons for the software VANTED (http://vanted.org) that implement fast stress minimization and a multilevel graph drawing framework respectively. The add-ons are to be developed for the class “Softwareprojekt” in the summer term 2019 at the University of Konstanz. This document lays out the requirements for the add-ons.

Scope

As already mentioned, the software is to be developed as add-ons for the VANTED system. The software shall be developed in the Java programming language. Two network layout methods for undirected graphs are to be added.
The stress minimization approach which tries to minimize a so-called stress function and the multilevel framework which is not an actual layout algorithm but a way to apply another layout algorithm at each level of a ‘hierarchy’ of simplified versions of the graph.

Abbreviations, Acronyms and Definitions

- **VANTED** refers to “Visualisation and Analysis of Networks containing Experimental Data”.
- **Add-on** refers to an extension of the VANTED software that can be loaded by VANTED and enhances it’s functionality.
- **OGDF** refers to “Open Graph Drawing Framework” (http://www.ogdf.net/doku.php)
- **GUI** refers to “Graphical User Interface”
- **MLF** refers to “Multilevel Framework”
- **SM** refers to “Stress Minimization”

References

- **[Spec]** Softwareprojekt2019_Gruppe2Thema.pdf
- **[V]** VANTED source code (https://bitbucket.org/vanted-dev/vanted/src/master/)
- **[Meet]** The initial meeting on the 25 April 2019
- **[Meet2]** The supervisor meeting on the 2 May 2019
- **[Eval]** Bartel2011_Chapter_AnExperimentalEvaluationOfMult.pdf

Overview

This document is divided into three chapters. The first chapter (which you are reading right now) gives an introduction and defines the purpose and scope of the software product. It also lists abbreviations and lists relevant definitions. The second chapter is a non-formalized overview of the software that is to be developed. The third chapter lists specific requirements and illustrates the system’s usage by means of a use-case diagram.

Overall description

Product Perspective

The layouters added by the add-ons can be used in the same way as existing layouters within VANTED.
The software shall function as specified by the requirements and shall be thoroughly tested and documented.

Product Functions

The add-ons enable users to layout graphs. They provide a settings dialogue where the user can set parameters for the respective layout method. The usage of the add-ons shall not require sophisticated knowledge of the underlying methods.
Roughly speaking, the two add-ons' functionality is as follows:

Multilevel Framework

The goal of the multilevel approach is to improve the performance and quality of energy-based layout algorithms by applying them at different 'levels' (coarsened versions of the graph). The multilevel framework works in three phases:

Coarsening

Build a sequence of increasingly simpler graphs by merging several vertices into one. The same is done on the coarsened graph again and again to create the 'levels' until a stop condition is reached.

Single Level Layout

On each level, a layout method (chosen by the user) is applied to the coarsened graph.

Placement

After the layout method has been applied on a level. The merged nodes need to be 'unmerged' and added back to the layout.

See [Eval] for an overview of the multilevel approach.

Stress Minimization

Stress minimization is a layout procedure. Here a stress function is used:

$\sum_{i<j} w_{ij}(d_{ij} - ||p_i - p_j||)^2$, where d_{ij} is the graph theoretical distance between the vertices v_i and v_j , w_{ij} is the weight usually defined as $\frac{1}{d_{ij}^2}$ and p_i and p_j are the positions of the vertices v_i and v_j . The Euclidean distance of all node pairs is compared with their theoretical

graph distance. Repeated estimation we approach the optimal value (stress of the graph is 0).

User Characteristics

The target group are users of the VANTED software.

Constraints

The stress minimization shall be fast (i.e. not asymptotically slower than existing implementations; see requirements for details). The software should be finished until 23 July 2019.

Assumptions and Dependencies

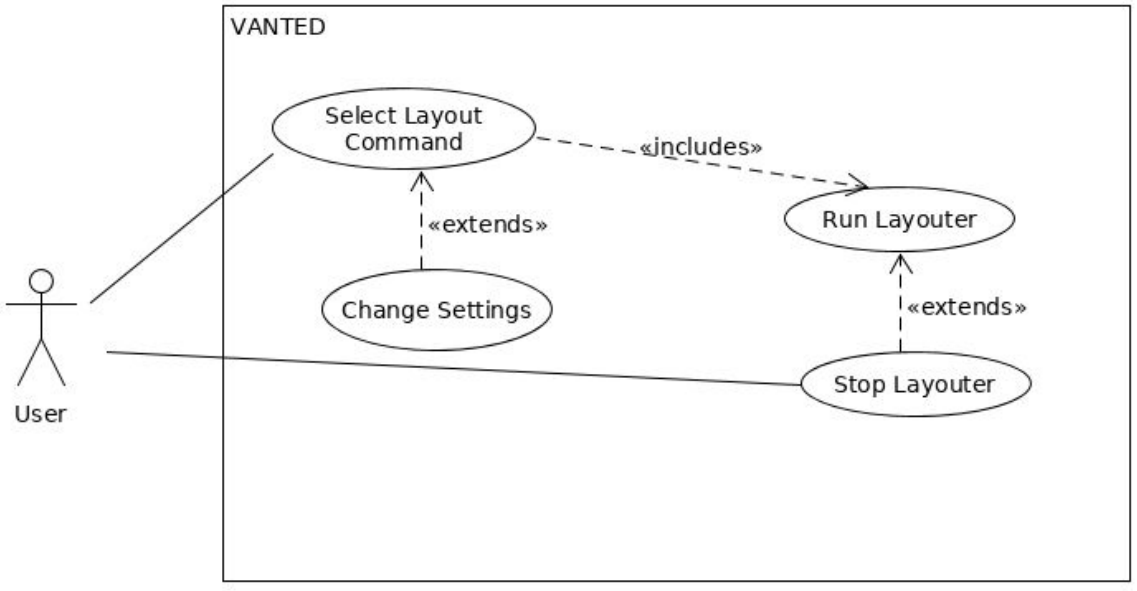
Users are already able to use the VANTED software and possess a computer that is able to run it (the computer has the complies to system requirements of VANTED [see http://www.vanted.org]). The add-ons depend on the VANTED software and Java Runtime Environment to run.

Specific Requirements

External Interfaces

The usage shall be similar to existing layouters already implemented in VANTED.

Use Case Diagram



Both layout methods will be used in the same way. First, the user selects the Layout Command in the “Layout” menu in VANTED. Then they can optionally change the settings and click on “Layout Network”. If the layouting process takes too long, they can click on “Stop Layout” to stop it.

Functions

R0: LayoutCommandStressMinimization
Function: The add-on shall add a layout command that performs stress minimization.
Description: The layout command shall be selectable from the layout tab in VANTED. For a more detailed description, see the section “Product Functions” above.
Source: [Spec]
Dependency: –

R1: LayoutCommandMultilevelFramework
Function: One add-on shall add a layout command that uses a multilevel approach for layouting.
Description: The layout command shall be selectable from the layout tab in VANTED. For a more detailed description, see the section “Product Functions” above.
Source: [Spec]
Dependency: –

R2: KindsOfGraphs
Function: The two layout commands shall work on undirected graphs.
Description: Both layout commands shall only work on undirected graphs. The multilevel framework shall additionally support working on graphs with edge weights.
Source: [Spec], [Meet2]
Dependency: R0, R1

R3: Dimension
Function: The resulting graph layout shall be displayed and calculated in two dimensions.
Description: –
Source: [Meet2]
Dependency: –

R4: DistanceMeasurement
Function: Distances calculated between the graph nodes shall be measured in Euclidean space.
Description: The stress minimization algorithm calculates the the distance between two nodes. This distance shall use the position of the nodes provided by VANTED and thus be Euclidean.
Source: [V], [Meet2]
Dependency: –

R5: MultilevelFrameworkAlgorithmSelection
Function: It shall be possible to change which layout algorithm the multilevel framework uses on each level.
Description: The specified algorithm will then be used for all levels. It shall not be possible to use different algorithms for different levels in the same execution of the multilevel layouter. Getting it to work with the force directed layout method has the highest priority.
Source: [Spec]
Dependency: R1

R6: Efficiency
Function: The layouters shall be implemented efficiently.
Description: The implementation of the multilevel shall be not be asymptotically slower than the "ogdf::ModularMultilevelMixer" implementation of OGDF and the implementation of the stress minimization layout shall not be asymptotically slower than the "ogdf::StressMinimization" implementation of OCDF. .
Source: [Spec], [Meet2]
Dependency: –

R7: JavaDoc
Function: All implemented methods (i.e. methods implemented by a member of team 2.2 as part of the two add-ons) shall be documented using JavaDoc.
Description: Methods such as getters and setters count as trivial and need not be documented if they exhibit their expected behavior.
Source: [Spec]
Dependency: –

R8: UserManual
Function: A short PDF user manual describing the usage of the layouters shall be written.
Description: The user manual shall describe how to use the two add-ons' GUI.
Source: [Spec]
Dependency: –

R9: StressMinimizationSettings
Function: The behavior of the stress minimization layouter shall be changable using user interface settings.
Description: –
Source: R0
Dependency: –

R10: UnitTests
Function: Every non-trivial implemented method (i.e. methods implemented by a member of team 2.2 as part of the two add-ons) shall have a corresponding unit test.
Description: –
Source: [Spec]
Dependency: –

R11: Benchmarks
Function: Networks of different size and characteristics shall be used to benchmark the layouters.
Description: Different networks of varying sizes and characteristics shall be used to benchmark the layouters and make sure that they work efficiently.
Source: [Spec]
Dependency: –

R12: ChangeSubAlgorithmParameters
Function: The parameters of the algorithm that the multilevel framework uses on all the levels shall be changeable.
Description: The settings interface of the algorithm shall be shown before the multilevel framework is run, allowing the user to change the parameters.
Source: –
Dependency: R1, R5

R13: MultilevelFrameworkCoarseningInterface
Function: There shall be an interface allowing the addition of further coarsening methods.
Description: It shall be possible to add new coarsening methods without changing the implementation of the multilevel framework itself.
Source: [Meet]
Dependency: R1

R14: MultilevelFrameworkPlacementInterface
Function: There shall be an interface allowing the addition of further placement methods.
Description: It shall be possible to add new placement methods without changing the implementation of the multilevel framework itself.
Source: –
Dependency: R1

R15: WorkOnSelection
Function: If nodes are selected the framework and algorithm shall only work with the induced subgraph (of the selected nodes).
Description: It shall be possible for the user to use the selection function of VANTED to select a specific subset of the nodes and let the framework and algorithm only operate on the induced subgraph created by selecting the nodes (and edges). Every node and edge not in this subgraph shall be ignored by the layouters.
Source: [V]
Dependency: –

R16: JavaVersion
Function: The add-ons shall be developed and tested using Java version 9.
Description: –
Source: [Meet2]
Dependency: –

R17: LayoutScaling
Function: The layout commands shall scale the graphs.
Description: The layout commands shall scale the graphs to ensure a proper display even when the graphs contain big nodes.
Source: [Meet2]
Dependency: –

R18: StopCommand
Function: It shall be possible to stop the layout methods before they are finished running.
Description: If the stress minimization algorithm is stopped, it shall use the current intermediate result as a layout. The multilevel framework shall restore the original layout upon being stopped.
Source: [Meet2]
Dependency: –

Project time schedule

Week/Final Date	**Event/Tasks**
29.4. - 5.5.	Read papers, Get development environment working
6.5. - 12.5.	Finish requirements document, discuss all papers, familiarize with vanted
13.5. - 19.5.	Decide on design, Write design specification
19.5.	Milestone 1: ● Design specification written
20.5. - 26.5.	● MLF: Implement and test data structures (for representing the coarsening levels), interfaces ● SM: finding and testing of useful libraries
27.5. - 2.6.	● MLF: Implement multilevel algorithm using a trivial merger and placement method ● SM: implementation
3.6. - 9.6.	● MLF: adjusting to VANTED and setting up force-directed ● SM: implementation
9.6.	Milestone 2: ● First prototype Multilevel Framework: Multilevel Framework working with Force-directed only using basic-level coarsening and placement. ● First prototype Stress Minimization via stress majorization
10.6. - 16.6.	● MLF: non-trivial merger ● SM: buffer if milestone 2 is not reached in time
17.6. - 23.6.	● MLF: non-trivial placement ● SM: bug fixing and polishing
24.6. - 30.6.	● MLF: benchmarks, adding options for layout-algorithms ● SM: benchmark and unit tests
1.7. - 7.7.	● MLF: User Manual and GUI/parameters ● SM: User Manual and GUI/parameters
7.7.	Milestone 3: ● Finished product ● Pass tests and benchmarks
8.7. - 14.7.	Polish project
15.7. - 21.7.	Time buffer, create final presentation
22.7. - 23.7.	Practise final presentation
23.7.	Final presentation