Spezifikation für OpenCAGE

Campus Software Solutions

Tobias Beichter
Dietrich Haak
Jörg Hoh
Hans Malte Kern
Stefan Mahler
Michael Reinsch
Michael Stürmer
Christian Viehmann
Fabian Wallwitz

Spezifikation für OpenCAGE

von Campus Software Solutions
Tobias Beichter
Dietrich Haak
Jörg Hoh
Hans Malte Kern
Stefan Mahler
Michael Reinsch
Michael Stürmer
Christian Viehmann
Fabian Wallwitz

Veröffentlicht 2001

Versionsgeschichte

Version 0.9 : 2001-01-11
Draft (internes Review)
Version 1.0 : 2001-01-18
Release
Version 1.1 : 2001-02-09
Release (Überarbeitung nach 1. Kunden-Review)
Version 1.2 : 2001-03-01 (Build: 1001330)
Release (Überarbeitung nach 2. Kunden-Review)

Inhaltsverzeichnis

Kapitel 0. Vorwort

Zweck des Dokumentes

Eine Software-Spezifikation dokumentiert die wesentlichen Anforderungen an ein bestimmtes, zu erstellendes Softwaresystem hinsichtlich Funktion und Qualität. Die Spezifikation beschreibt dabei in einer möglichst lösungsunabhängigen Form, was das System leisten soll. Dazu wird das Verhalten der Software in möglichst kleine Einheiten gegliedert und aus der Sicht des jeweiligen Benutzers beschrieben. Details der Realisierung bleiben bewusst außen vor, um nachfolgende Entwurfsentscheidungen nicht einzuschränken.

Im Rahmen des OpenCAGE-Projekts stellt das vorliegende Dokument das Ergebnis der abgeschlossenen Analyse- und Definitionsphase dar. Es bildet eine verbindliche Grundlage für alle folgenden Projektphasen und ermöglicht dem Kunden die Überprüfung der Analyseergebnisse.

Beschreibung des Projekts OpenCAGE

OpenCAGE (Open Computer Aided Graph Editor) wird von der Projektgruppe BaBu der Campus Software Solutions für den (fiktiven) Auftraggeber Fictitious Bauhaus Reengineering entwickelt. Es soll den Grapheneditor Rigi ersetzen, der im Rahmen des Bauhaus-Projekts der Fakultät Informatik der Universität Stuttgart verwendet wird. Rigi soll wegen seiner unzureichenden Geschwindigkeit, der schlechten Erweiterbarkeit und der gewachsenen und inkonsistenten GUI abgelöst werden. In der Entwicklung von OpenCAGE sollen diese Punkte besonders berücksichtigt werden.

Da es sich hierbei um ein Studienprojekt handelt, wird außerdem sehr auf eine durchdachte und solide Projektplanung sowie die konsequente Anwendung von softwaretechnischen Methoden Wert gelegt.

Projektziele

Das vordringlichste Projektziel ist, einen offenen und leicht erweiterbaren Grapheneditor für die Anbindung an die Bauhaus-Suite zu entwickeln. Eine einfache Änderbarkeit ermöglicht es auch externen Entwicklern, die Software an neue Anforderungen anzupassen und neue Funktionalität mit relativ geringem Aufwand hinzufügen zu können.

Da das Projekt in einer akademischen Umgebung stattfindet, ist die Wissensgewinnung sowie das Praktizieren und Vertiefen des gelehrten Stoffes ebenfalls ein nicht unerhebliches Ziel dieses Projekts, so dass auch ein Lerneffekt für die beteiligten Entwickler erwartet wird.

Andere Dokumente

Im Rahmen des Projekts OpenCAGE werden noch andere Dokumente entstehen, die im Projektplan detailliert beschrieben sind. Im Zusammenhang mit der Spezifikation ist besonders das *Begriffslexikon* zu nennen (welches sowohl als separates Dokument erhältlich, als auch im Anhang B zu finden ist), in dem sämtliche Begriffe definiert werden, die innerhalb dieser Spezifikation verwendet werden, und die in diesem Kontext mehrdeutig sein könnten.

Referenzen

Die Spezifikation basiert auf folgenden Dokumenten:

- [Eisenbarth, Thomas] GropiusSE: eine Ressource Flow Graph Bibliothek in Ada95 für das Speichern und Aufbereiten von Reengineeringinformationen.
- Protokolle der Kundengespräche
- Projektplan OpenCAGE-Projekt, Campus Software Solutions
- Angebote der (konkurrierenden) Gruppen

Zielgruppe der Spezifikation

Dieses Dokument richtet sich an 4 Gruppen:

- An die Entwickler, die dieses Dokument als die detaillierte Ausarbeitung der Kundenanforderungen betrachten, und es zum Nachschlagen benutzen, wenn im weiteren Verlauf des Projekts Unklarheiten über bestimmte Details der GUI, der funktionalen- oder nicht-funktionalen Anforderungen entstehen. Weiterhin werden anhand der Spezifikation das Benutzerhandbuch, die Online-Hilfe und Testfälle erstellt.
- An den Kunden, der anhand dieser Spezifikation die Endabnahme durchführt, und die beschriebenen Anforderungen mit den Möglichkeiten und Fähigkeiten des fertigen Programms vergleicht.
- An die Personen, die Reviews im Rahmen von OpenCAGE durchführen.
- An interessierte Dritte, die sich anhand dieser Spezifikation ein Bild über OpenCAGE machen wollen.

Grobgliederung der Spezifikation

Die Spezifikation gliedert sich in folgende Teile:

- *Vorwort* (Kapitel 0)
- *Nicht-funktionale Anforderungen* (Kapitel 1)
- *Allgemeine funktionale Anforderungen* (Kapitel 2)
- *GUI - Konzepte* (Kapitel 3)
- *GUI - Realisierung* (Kapitel 4)
- *Use Cases* (Kapitel 5)
- *Begriffslexikon* (Anhang B)
- *Glossary* (Anhang C)

Versionen dieses Dokuments

Versionsgeschichte

Version 0.9 : 2001-01-11
Draft (internes Review)
Version 1.0 : 2001-01-18
Release
Version 1.1 : 2001-02-09
Release (Überarbeitung nach 1. Kunden-Review)
Version 1.2 : 2001-03-01 (Build: 1001330)
Release (Überarbeitung nach 2. Kunden-Review)

Kapitel 1. Nicht-funktionale Anforderungen

Leistungsanforderungen

Mengengerüst

Typischerweise haben die zu bearbeitenden Graphen bis zu 100.000 Knoten und eine Million Kanten. Es gibt keine feste Obergrenze für die Menge der Knoten und Kanten.

Laufzeit

Mit den oben genannten typischen Datenmengen wird ein flüssiges Arbeiten garantiert, d.h. es gelten im Single-User-Betrieb auf den unter *Verbrauchsverhalten* (Abschnitt namens *Verbrauchsverhalten*) genannten Rechnertypen Reaktionszeiten von unter drei Sekunden. Je mehr Daten OpenCAGE zu visualisieren und verwalten hat, desto länger werden die Reaktionszeiten. Werden die typischen Datenmengen überstiegen, können wesentlich längere Reaktionszeiten die Folge sein. Die Laufzeiten werden linear skalieren.

Für alle Funktionen, die über *Schnittstellen zu externen OpenCAGE-Funktionen* (Abschnitt namens *Schnittstellen zu externen OpenCAGE-Funktionen*) angesprochen werden, existieren keine obere Grenze der Laufzeit.

Entwurfseinschränkungen

Folgende Einschränkungen wurden von Fictitious Bauhaus Reengineering vorgegeben:

- Ada95 als Implementierungssprache
- Die Verwendung von GtkAda als GUI-Bibliothek
- Die Benutzung der RFG-Bibliothek als Schnittstelle zum Bauhaus-Projekt
- Die Entwurfs- und Dokumentationssprache ist Englisch.
- Die Spezifikations- und Entwurfsdokumente werden im DocBook-Format erstellt.

Verfügbarkeit

OpenCAGE wird nur im normalen Tagesbetrieb laufen, d.h. am Ende jedes Arbeitstages wird OpenCAGE beendet. Innerhalb dieser Zeit dürfen keine Abstürze oder sonstige Arten von Datenverlust auftreten, die unmittelbar durch OpenCAGE verursacht werden. OpenCAGE soll keine Ressource-Leaks aufweisen, die die Arbeit mit dem Rechner beeinflussen.

Robustheit

OpenCAGE soll gegenüber Lastsituationen des Systems, auf dem es in Betrieb ist, robust sein. Große Datenmengen, die innerhalb von OpenCAGE verarbeitet werden (mehr als die typischen 100.000 Knoten und eine Million Kanten), dürfen OpenCAGE nicht zum Absturz oder sonstigen Verweigerungen der spezifizierten Funktionen bringen, sofern OpenCAGE genügend Systemressourcen zur Verfügung stehen.

Eingaben eines Benutzers, die innerhalb eines bestimmten Kontextes nicht erlaubt sind, werden abgefangen. Dies geschieht durch Ausgrauen des Bestätigungsbuttons bzw. des Menüpunkts, so dass die Funktion nicht ausgeführt werden kann. Ist dies nicht einfach möglich, wird der Benutzer nach dem Ausführen der Funktion mit einer entsprechenden Meldung auf das Problem hingewiesen und die Funktion abgebrochen.

Portabilität

Zwischen den Betriebssystemen SunOS und GNU/Linux wird eine Quellcode-Kompatibilität bestehen. Für Windows muss diese konzeptbedingt nicht unbedingt gelten. Es wird darauf Wert gelegt, dass die Portabilität auf andere Plattformen durch den Systementwurf nicht beschränkt wird.

Schnittstellen zu anderen externen Systemen

Explizite Schnittstellen sind nur zur RFG-Bibliothek vorgesehen.

Zu folgenden Systemen sind implizite Schnittstellen vorhanden:

- Betriebssystem; diese Schnittstelle soll durch geschickte Kapselung möglichst übersichtlich und einfach gehalten werden.
- Graphische Oberfläche (GtkAda)

Schnittstellen zu externen OpenCAGE-Funktionen

Ein wichtiger Punkt ist die einfache Erweiterbarkeit von OpenCAGE. Dies soll vor allem durch offene, dokumentierte Schnittstellen zum Aufruf externer Funktionen gewährleistet werden. Diese sind als Plugins konzipiert, sofern dies von den verschiedenen Plattformen unterstützt wird.

Die folgenden internen Schnittstellen werden mindestens in OpenCAGE enthalten sein. Eine genaue Dokumentation der Schnittstellen als auch das Design aller Schnittstellen wird im Systementwurf enthalten sein.

Analyse-Schnittstelle

Eine asynchrone Schnittstelle zu externen Analysefunktionen. Diese unterstützt folgende Merkmale: Ablauf im Hintergrund, Abbruch der Analyse, Vorbedingungsprüfung, Statusmeldungen, Analysen-spezifische Parameterdialoge und eine einfache Handhabung mehrerer vordefinierter Parametersätze.

Layout-Schnittstelle

Eine asynchrone Schnittstelle zu externen Layoutfunktionen. Diese unterstützt folgende Merkmale: Ablauf im Hintergrund und Abbruch der Layoutberechnungen.

Metrik-Schnittstelle

Eine asynchrone Schnittstelle zu externen Metrikberechnungen. Diese unterstützt folgende Merkmale: Ablauf im Hintergrund und Abbruch der Metrikberechnungen.

Export-Schnittstelle

Eine Schnittstelle zu externen Exportfiltern.

Filter-Schnittstelle

Eine Schnittstelle zu externen Filteroperationen.

Operations-Schnittstelle

Eine Schnittstelle zu externen Programmoperationen.

Anforderungen an die Dokumentation

Dokumente werden meistens mit Hilfe von DocBook Version 3.1 erstellt. Daher werden die Spezifikation, der Entwurf, das Benutzerhandbuch und die Online-Hilfe in diesem Format erstellt. Alle anderen Dokumente sind entweder in DocBook, LaTeX oder ASCII 7bit zu erstellen. Einzelheiten legt der Projektplan fest.

Die bevorzugte Sprache ist Englisch. Daher werden der Entwurf, das Benutzerhandbuch, die Online-Hilfe und das Begriffslexikon in englischer Sprache gehalten sein. Alle anderen Dokumente sind entweder in englischer oder deutscher Sprache verfasst. Die Kommentare innerhalb des Source-Codes werden in Englisch gehalten. Einzelheiten legt der Projektplan fest.

Abnahmekriterien

Die Abnahme-Kriterien sind zum jetzigen Zeitpunkt noch nicht definiert.

Qualitätsmerkmale

In diesem Abschnitt wird die Produktqualität beschrieben. Hier ist definiert, wie gut OpenCAGE für den Benutzer und die (zukünftigen) Entwickler sein soll.

Zur Beschreibung der Qualitätsmerkmale der Software halten wir uns an die DIN ISO 9126 (*DIN ISO 9126: Informationstechnik - Beurteilen von Softwareprodukten, Qualitätsmerkmale und Leitfaden zu deren Verwendung, Beuth, 1991*), denn in diesem Qualitätsmodell sind die Überschneidungen der Qualitätsmerkmale nur minimal.

Wir definieren nicht nur die Ziele für das Produkt selbst, sondern auch für den Prozess, in dem das Produkt entsteht. Die Qualität von OpenCAGE hängt im wesentlichen von der Qualität des Prozesses ab, in dem es entsteht.

Die Ziele des Projektes wie die Einhaltung von Terminen oder Kosten sowie weiterer Projektattribute sind im Projektplan beschrieben.

Änderbarkeit

Eine gute Architektur und gute Dokumentation bestimmen die Erweiterbarkeit von OpenCAGE. Die Güte von Architektur und Dokumentation werden durch Qualitätsmerkmale bestimmt. Dieser Abschnitt beschreibt diese Qualitätsmerkmale und den Aufwand für Dritte, welche die entstehende Software warten und erweitern werden. Dies schließt Änderungen, Korrekturen und Verbesserungen ein. Wartung kann durch Anpassungen an Änderungen der Umgebung oder geänderte Anforderungen der funktionale Spezifikation nötig sein.

Analysierbarkeit

Der Aufwand, änderungsbedürftige Teile zu bestimmen oder Ursachen möglicher Mängel diagnostizieren zu können, soll so gering wie möglich sein. Dies wird vor allem durch eine *umfassende, hochwertige Dokumentation* erreicht. Dies umfasst die Dokumentation der Anforderungen, der Lösungsstrategien, der Entwurfsideen sowie der Architektur bzw. des Designs. Ebenso wichtig ist deren stetige Aktualisierung. Zusätzlich wird der Quellcode ausführlich dokumentiert und der Testplan sowie ein Testprotokoll erstellt. Das Design von OpenCAGE ist auf Zuwachs ausgelegt. Dies bedeutet, dass bereits bei der Planung der Software und des Entwurfs dafür Vorsorge getroffen wird, und daher die Architektur klar strukturiert und einzelne Komponenten sauber getrennt werden. Die einzelnen Komponenten von OpenCAGE kommunizieren über einheitliche und wohldefinierte Schnittstellen, die ausführlich dokumentiert und offen gelegt sind.

Modifizierbarkeit

Der Aufwand zur Ausführung von Wartung ist so gering wie möglich zu halten. Um dies zu erreichen, werden alle Teile des Quellcodes übersichtlich strukturiert und selbsterklärende Bezeichner verwendet. Um eine einheitliche Strukturierung und Dokumentation des Quellcodes sicherzustellen wird der vom Kunden gewünschte Programmierstyleguide (*ADA 95 QUALITY AND STYLE: GUIDELINES FOR PROFESSIONAL PROGRAMMERS, SPC-94093-CMC, Version 01.00.10, October 1995, http://adaic.org/docs/styleguide/95style/*, Kap. 2 und 3) Anwendung finden.

Stabilität

Stabilität bedeutet hier, dass bei Änderungen der Software möglichst nur an geänderten Komponenten Effekte auftreten sollen und alle anderen Teile unempfindlich sind. Nur ein guter Entwurf kann die Wahrscheinlichkeit des Auftretens unerwarteter Wechselwirkungen bei Änderungen minimieren. Erreicht wird dies durch hohe Lokalität, die die Reichweite von Änderungen auf die modifizierte Komponente einschränkt. Solche Änderungen werden durch das Testprotokoll für OpenCAGE unterstützt.

Prüfbarkeit

Der Aufwand, der zur Prüfung der geänderten Software notwendig ist, soll so gering wie möglich sein. Der Kunde bekommt alle entworfenen Testfälle sowie alle Testprotokolle und muss somit nur den Test der von ihm vorgenommenen Erweiterungen, sowie der davon betroffenen Teile von OpenCAGE, planen.

Übertragbarkeit

Dieser Abschnitt beschreibt die Eignung der Software, von einer Umgebung in eine andere übertragen zu werden. Dies kann eine andere Hardware- bzw. Softwareumgebung oder eine andere organisatorische Umgebung sein.

Anpassbarkeit

OpenCAGE kann in verschiedenen Umgebungen genutzt werden, solange die Anforderungen an die externe Umgebung erfüllt sind. Dies sind

- es existiert ein Ada95-Compiler
- es existiert eine Portierung der GtkAda-Bibliothek

auf dem Zielsystem. Dies wird durch folgende Maßnahmen ermöglicht:

- Plattformunabhängige Realisierung von OpenCAGE durch die Verwendung von Ada95.
- möglichst keine Verwendung plattformspezifischer Möglichkeiten und Bibliotheken.

Installierbarkeit

Der Aufwand zum Installieren von OpenCAGE soll so gering wie möglich sein. Bei der Erstellung des Programms wird auf möglichst hohe Plattformunabhängigkeit Wert gelegt, siehe *Anpassbarkeit* (Abschnitt namens *Anpassbarkeit*). Um dies zu gewährleisten, wird ein kleines Installierungstool mit bereitgestellt (z.B. make). So ist es möglich, OpenCAGE komplett ohne große Probleme auf jedem Zielsystem, für das es zusätzlich das Programm GNU make gibt, zu compilieren, zu linken und zu installieren.

Konformität

OpenCAGE ist konform zu folgenden Normen:

- Styleguide: *ADA 95 QUALITY AND STYLE: GUIDELINES FOR PROFESSIONAL PROGRAMMERS, SPC-94093-CMC, Version 01.00.10, October 1995, http://adaic.org/docs/styleguide/95style/*, Kap. 2 und 3
- Dokumentation in SGML
- Ada95

Austauschbarkeit

OpenCAGE ist eine Neuentwicklung, die eine bereits existierende Lösung ersetzen soll. Ansonsten ist nicht vorgesehen, dass eine andere Anwendung durch OpenCAGE ersetzt werden kann. Die Hauptkomponenten (GUI, API und Schnittstellen) von OpenCAGE werden so unabhängig wie möglich entworfen, um einen späteren Austausch der einzelnen Komponenten zu ermöglichen.

Benutzbarkeit

Wie hoch ist der Aufwand, der zur Benutzung erforderlich ist? Die Zielgruppe dieser Anwendung sind anfangs Experten aus der Universität. OpenCAGE soll jedoch auch von Anwendern eingesetzt werden, die nur über Erfahrung bei der Bedienung gängiger Softwareprodukte verfügen. Um dieser Zielgruppe gerecht zu werden, orientiert sich OpenCAGE an der Bildschirmverarbeitungsverordnung bzw. den Ergonomierichtlinien nach DIN EN ISO 9241 Teil 10 - „Gestaltungsgrundsätze zur Dialoggestaltung“. Diese geben einen Orientierungsrahmen für die Entwicklung von graphischen Benutzungsoberflächen, der folgende Punkte beinhaltet:

Aufgabenangemessenheit

Das GUI von OpenCAGE soll den Benutzer bei der Erledigung seiner Aufgabe unterstützen, ohne ihn durch andere Eigenschaften von OpenCAGE zu belasten.

Selbstbeschreibungsfähigkeit

Dem Benutzer soll Leistungsumfang und Einsatzzweck jedes Elements des GUI unmittelbar verständlich sein bzw. auf Verlangen des Benutzers sollen diese Informationen erläutert werden.

Steuerbarkeit

Der Benutzer soll die Geschwindigkeit des Ablaufs und die Reihenfolge der Eingabe von Informationen beeinflussen können.

Erwartungskonformität

Das GUI von OpenCAGE soll sich so verhalten, wie der Benutzer es aufgrund des Benutzerhandbuchs bzw. der Online-Hilfe, der Benutzerschulung und der bisherigen Arbeit mit OpenCAGE und anderen GUIs erwartet.

Fehlertoleranz

Bei Fehleingaben kann der Benutzer mit keinem oder nur geringem Korrekturaufwand das gewünschte Arbeitsergebnis erzielen. Dazu müssen dem Benutzer die Fehler verdeutlicht werden.

Individualisierbarkeit

Das GUI von OpenCAGE sollte Anpassungen an die Erfordernisse der Aufgabe und Vorlieben des Benutzers zulassen.

Lernförderlichkeit

Der Benutzer sollte beim Erlernen des GUI von selbigem unterstützt und angeleitet werden.

Funktionalität

Die Software besitzt Funktionen, um die definierten Anforderungen zu erfüllen, wie sie mittels der UML-Notation in den *Use Cases* (Kapitel 5) beschrieben sind. Einige Zusatzfunktionen von OpenCAGE, die die Benutzerfreundlichkeit und Interaktionsmöglichkeiten weiter verbessern würden, werden im Rahmen dieses Projektes nicht implementiert da sonst der Gesamtaufwand und die Entwicklungsdauer zu hoch wären.

Richtigkeit

OpenCAGE soll genau die Operationen durchführen, die der Benutzer, mittels der ihm gegebenen Interaktionsmöglichkeiten, ausgeführt haben will.

Interoperabilität

Die Fähigkeit mit anderen System zusammenzuarbeiten ist im Fall von OpenCAGE nicht wichtig. Es muss lediglich auf Funktionen der RFG-Bibliothek arbeiten.

Sicherheit

Da der Kunde von einer gutartigen Umgebung ausgeht und es sich bei den Daten, die mit OpenCAGE bearbeitet werden, nicht um sicherheitsrelevante Daten handelt, sind keine Sicherheitseinstellungen außer dem Read-Only-Mode (siehe *Zustände*, Abschnitt namens *Zustände* in Kapitel 3) vorgesehen.

Effizienz

Dieser Abschnitt beschreibt das Verhältnis von Leistungsniveau der Software und Umfang der eingesetzten Betriebsmittel.

- Wie schnell muss OpenCAGE Benutzerinteraktionen bearbeiten?
- Wieviele Benutzer können OpenCAGE (gleichzeitig) nutzen?

Zeitverhalten

Der Kunde verlangt nach besonderen Anstrengungen, um die Antwort- und Verarbeitungszeit in engen Grenzen zu halten. Genaue Daten sind unter *Leistungsanforderungen* (Abschnitt namens *Leistungsanforderungen*) zu finden.

Wieviele Benutzer können mit OpenCAGE gleichzeitig arbeiten? OpenCAGE ist als ein Ein-Benutzer-System konzipiert, da eine Mehr-Benutzer-Lösung eine massive Änderung des RFGs zur Folge hätte.

Verbrauchsverhalten

In der folgenden Tabelle sind die Betriebsmittel aufgeführt, die abrufbar sein müssen, um die geforderten Funktionen erfüllen zu können:

Tabelle 1-1. Übersicht über die benötigten Betriebsmittel

System	Solaris	GNU/Linux	Windows
Betriebssystem	SUN OS 5.1	Kernel > 2.2.12	Windows NT 4.0
Hardware	Ultra Sparc 10	Pentium-III-Klasse, mind. 256 MB RAM	
Beispielsystem	storm	gspc32	

Zuverlässigkeit

Welches Leistungsniveau muss unter festgelegten Bedingungen über einen festgelegten Zeitraum aufrecht erhalten werden?

Reife

Die Reife soll so hoch sein, dass OpenCAGE im vollen Umfang für die spezifizierten Aufgaben verwendet werden kann. Die Versagenshäufigkeit durch Fehlzustände soll so gering wie möglich gehalten werden.

Fehlertoleranz

Das Leistungsniveau kann bei einigen (unerwarteten) Softwarefehlern eingehalten werden. Die Fehlertoleranz ist aber nicht besonders hoch, da trotzdem Ausnahmesituationen auftreten können, in denen das Verhalten von OpenCAGE nicht definiert ist.

- Fehler, die vom Betriebssystem oder anderen verwendeten Bibliotheken kommen, werden, soweit möglich, abgefangen und dem Benutzer angezeigt. Mit Hilfe des Benutzers wird versucht, den Fehler zu beheben und den normalen Programmablauf aufrechtzuerhalten.
- Bei Fehlern des Betriebssystems oder verwendeter Bibliotheken, die nicht behoben werden können, wird versucht, OpenCAGE kontrolliert zu beenden und alle ungesicherten Daten zu sichern. Ob dies gelingt, hängt von der Art des Fehlers ab.

- Bei Totalabstürzen des Betriebssystems oder verwendeter Bibliotheken ist das Verhalten undefiniert.

Wiederherstellbarkeit

Unter diesem Punkt versteht man die Fähigkeit von OpenCAGE, Daten bei Versagen wiedergewinnen zu können. OpenCAGE selbst sieht hier keine Möglichkeit vor, Daten wiederherstellen zu können, sollte der RFG oder die Konfiguration von OpenCAGE beschädigt werden. Bei Verlust der Konfiguration verwendet OpenCAGE die Standardeinstellungen und die ursprünglichen Daten sind verloren. Allerdings kann durch die Verwendung anderer Systeme, z.B. durch Backups usw., die Wiederherstellung verbessert werden.

Kapitel 2. Allgemeine funktionale Anforderungen

Programmkomponenten

OpenCAGE besteht aus folgenden Programmkomponenten:

- *Workbench* (Abschnitt namens *Workbench* in Kapitel 3): Sie dient zur allgemeinen Administration.
- *Master* (Abschnitt namens *Master* in Kapitel 3):
  - *Window-Master* (Abschnitt namens *Window-Master* in Kapitel 3): Er wird zum Fenster-Management verwendet.
  - *View-Master* (Abschnitt namens *View-Master* in Kapitel 3): Er dient zur Anwendung von Operationen, die eine komplette View betreffen.
  - *Mode-Master* (Abschnitt namens *Mode-Master* in Kapitel 3): Diese dienen dazu, um komplexere Operationen auf Graph-Areas auszuführen.
- *Graph-Window* (Abschnitt namens *Graph-Window* in Kapitel 3): Sie enthalten die Präsentation von RFG-Teilen und sind somit von besonderer Bedeutung für OpenCAGE. Für jedes Graph-Window existiert ein Eintrag im Window-Master.
- weitere Fenster: Es kann zu jedem Zeitpunkt der Hilfebrowser oder der Editor, der Sourcecode anzeigt, geöffnet sein.

Erzeugen einer neuen View

Das Erzeugen einer neuen View impliziert:

- OpenCAGE erzeugt einen neuen Eintrag im View-Master.

Erzeugen eines neuen Graph-Windows

Das Erzeugen eines neuen Graph-Windows impliziert:

- OpenCAGE erzeugt einen neuen Eintrag im Window-Master. Dabei wird dieser Fenstereintrag an den Grapheneintrag angehängt, den es repräsentiert. Existiert dieser Eintrag im Window-Master noch nicht, so wird erst dieser erzeugt.
- Das neue Graph-Window wird geöffnet.

Basis-Views

Trashcan-View

Die Trashcan-View ist eine besondere View. Sie ermöglicht es, gelöschte RFG-Objekte wiederherzustellen. Wenn aus einer View RFG-Objekte gelöscht werden, so werden diese in die Trashcan-View verschoben.

Durch *Undelete* (Abschnitt namens *Undelete* in Kapitel 5) kann der Benutzer die von ihm gelöschten RFG-Objekte aus der Trashcan-View wieder in ihre ursprüngliche View zurückverschieben.

Durch *Clear Trashcan* (Abschnitt namens *Clear Trashcan* in Kapitel 5) kann der Benutzer alle RFG-Objekte aus der Trashcan-View entfernen.

Die RFG-Objekte in der Trashcan-View werden zudem automatisch von OpenCAGE entfernt, bevor

a. eine Löschoperation durchgeführt wird,

b. eine RFG-Manipulationsoperation auf der View ausgeführt wird, aus der RFG-Objekte, welche sich derzeit in der Trashcan-View befinden, stammen. Dies soll möglichen Inkonsistenzen bei einer Undelete-Operation vorbeugen.

Weitere RFG-Manipulationsoperationen kann der Benutzer auf die Trashcan-View nicht anwenden.

Die Clipboard-View

Die Clipboard-View ist ebenfalls eine besondere View. Sie ermöglicht es, RFG-Objekte zwischenzuspeichern und so diese z.B. von einer View in eine andere zu kopieren oder zu verschieben.

Copy (Abschnitt namens *Copy* in Kapitel 5) ⇒ Kopieren in die Clipboard-View

Cut (Abschnitt namens *Cut* in Kapitel 5) ⇒ Verschieben in die Clipboard-View

Paste (Abschnitt namens *Paste* in Kapitel 5) ⇒ Kopieren aus der Clipboard-View in eine andere View

Die Clipboard-View kann lediglich die RFG-Objekte der letzten Copy- oder Cut-Operation aufbewahren. Wird also eine Copy- oder Cut-Operation ausgeführt, werden alle bereits in der Clipboard-View enthaltenen RFG-Objekte automatisch aus ihr entfernt.

Weitere RFG-Manipulationsoperationen kann der Benutzer auf die Clipboard-View nicht anwenden.

Die Basis-Views des RFGs

Folgende Views sind derzeit fest im RFG vorgegeben und stehen ständig zur Verfügung:

- Call-View
- Dataflow-View
- Enviroment-View

- File-View
- Type-View

Diese Views sind immer gesperrt.

Skript-Funktionalität

Jede Funktion, die der Benutzer innerhalb von OpenCAGE aufgerufen wird, wird in einer für jeden Benutzer eindeutigen Protokolldatei mitgeschrieben. Diese Datei kann dazu genutzt werden, um einen kompletten Programmlauf nachzuvollziehen, beispielsweise für Debugging-Zwecke.

Für jedes Projekt, das der Benutzer öffnet, wird eine separate Protokolldatei angelegt. Der Name dieser Datei ergibt sich aus dem Namen der RFG-Datei mit der Endung `.ocs`. Bereits existierende Protokolldateien werden bei jedem Öffnen des Projekts gelöscht und neu angelegt.

Sobald eine RFG-Datei geöffnet wird, schreibt OpenCAGE jede Veränderung durch den Benutzer (z.B. Knoten verschieben) oder durch OpenCAGE (z.B. Analyse) in der Protokolldatei mit.

Ausführen einer Protokolldatei

Die Protokolldatei kann über die Menüleiste der *Workbench* (Abschnitt namens *Workbench* in Kapitel 4) ausgeführt werden (siehe auch Use-Case *Run Script*, Abschnitt namens *Run Script* in Kapitel 5).

Das Skript wird Befehl für Befehl ausgeführt. Ist das Skript auf die aktuelle Präsentation des momentanen RFG nicht anwendbar, wird eine Fehlermeldung in der Error-Log-Konsole ausgegeben. Die Abarbeitung des Skripts wird unterbrochen.

Das Format der Protokolldatei

Das Format, in dem die Befehle innerhalb der Protokolldatei gespeichert sind, wird erst während der Entwurfsphase festgelegt und dokumentiert.

Einstiegspunkte für die Use-Cases

Der Einstiegspunkt Skript

Neben der Möglichkeit, Use-Cases interaktiv auszuführen, ist bei OpenCAGE eine Skriptfunktion als Erweiterung vorgesehen. Sie wird durch Kommandos, die aus einer Protokolldatei eingelesen werden, realisiert (siehe *Skript-Funktionalität*, Abschnitt namens *Skript-Funktionalität*).

Die entsprechenden Use-Cases erhalten als zusätzlichen Einstiegspunkt „Skript“. Die Eingabe erfolgt somit nicht mehr über Maus und Tastatur, sondern die entsprechenden Parameter werden aus der Skript-Datei eingelesen. Damit sind die Punkte, die die Benutzereingabe betreffen, für diesen Fall irrelevant.

Der Einstiegspunkt Direkte Manipulation

Unter der direkten Manipulation versteht man das Anklicken oder das Verschieben eines graphischen Obkekts mit der Maus (z.B. Anklicken eines Knotens). Bei der direkten Manipulation wird der Mauszeiger über das zu manipulierende Objekt bewegt und dort die Funktion ausgeführt.

Zur direkten Manipulation gehört auch das Selektieren von Knoten. In den Use-Cases gilt oft, dass mindestens ein Knoten selektiert sein muss.

Einstiegspunkt Kontextmenüs

Der View-Master, Window-Master und die Graph-Windows haben Kontextmenüs (siehe *Kontextmenüs*, Abschnitt namens *Kontextmenüs* in Kapitel 3). Diese werden angezeigt, wenn im Fenster die rechte Maustaste gedrückt wird. Danach kann der Benutzer eine Aktion auswählen.

Graph-Window-Buttons

Jedes Graph-Window hat direkt oberhalb der vertikalen Scrollbar Buttons. Diese sind dazu gedacht, alle RFG-Objekte im selben Graph-Window zu selektieren bzw. die Selektion zu invertieren.

Menüleiste

Die Menüleiste befindet sich in der Workbench.

Der Programmstart

OpenCAGE kann direkt oder mit dem Parameter der zu öffnenden RFG-Datei gestartet werden: **opencage** *`rfg-filename`*.

OpenCAGE lädt die Programmparameter (siehe *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien*, Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien*) aus dem Verzeichnis `~/.opencage/`.

Falls der Benutzer beim Programmstart als Parameter eine RFG-Datei `projekt.rfg` angegeben hat, werden die Projektparameter aus der Datei `projekt.ocp`, die im selben Verzeichnis liegt, geladen und dementsprechend sämtliche Graph-Windows mit Inhalt erzeugt, sowie der View- und Window-Master mit den entsprechenden Daten der Graph-Windows und Views angezeigt.

Falls die benutzerdefinierten Konfigurationsdateien nicht gefunden werden, lädt OpenCAGE die Standardeinstellungen aus dem entsprechenden Verzeichnis (z.B. `/etc/opencage/` unter Unix). Sollte dies nicht erfolgreich sein, verwendet OpenCAGE die einkompilierten Standardeinstellungen.

Der Benutzer kann anschließend mit OpenCAGE arbeiten.

Programm-, Projektspezifische Einstellungen und Konfigurationsdateien

Es gibt programm- und projektabhängige Parameter. Diese werden entsprechend in Dateien abgespeichert. Das Format, in dem die Einstellungen gespeichert sind, sowie die Aufteilung der Einstellungen über die einzelnen Dateien werden in einem späteren Dokument beschrieben.

Programmspezifische Einstellungen

Die Standard-Konfigurationsdateien liegen unter `/etc/opencage/` und enthalten die Werte, die beim Fehlen einer benutzerdefinierten Einstellung geladen werden. Alle benutzerdefinierten Einstellungen überschreiben die Standard-Einstellungen. Ein Nachladen der Parameter ist möglich. Folgende Einstellungen sind in den Standard-Konfigurationsdateien enthalten.

- Standard-Style
- Standard-Fensterparametersatz
- Pfad und Dateiname des Editors
- Pfad und Dateiname des Browsers
- Sprache (wird nicht implementiert); der Default-Wert ist Englisch.
- Konfiguration der Symbolleiste
- Konfiguration der Metrikparameter und deren Visualisierung
- Shortcuts, Hotkeys für die Operationen
- Hintergrundfarbe des View-Masters und Window-Masters
- Schriftfarbe der Fenstereinträge und Grapheneinträge im Window-Master.
- Die Dateinamen der Pixmaps fuer die Knoten (siehe *Knoten*, Abschnitt namens *Knoten* in Kapitel 3)
- Visualisierte Teile der Knotennamen (siehe *Knoten*, Abschnitt namens *Knoten* in Kapitel 3)
- Einschalten der Backup-Funktion
- Zoom-Grenzen

Die benutzerdefinierten Einstellungen liegen unter `~/.opencage/` und besitzen dasselbe Format und Aufteilung wie die globalen Einstellungen.

Die benutzerdefinierten Einstellungen werden den Standardeinstellungen vorgezogen; wenn sie nicht vorhanden sind, werden die Standardeinstellungen geladen und benutzt. Sind nur teilweise benutzerspezifische Einstellungen vorhanden, so werden diese verwendet und fuer den Rest die Standardeinstellungen genutzt.

Projektspezifische Einstellungen

Projektspezifische Parameter sind Einstellungen, die nur für ein spezielles Projekt gelten. Die projektabhängigen Daten werden in einer Projektdatei im gleichen Verzeichnis wie die RFG-Datei mit der Endung `.ocp` abgespeichert. Bei jedem Speichern des RFGs werden diese Einstellungen gespeichert. Beim Öffnen einer RFG-Datei wird versucht, die entsprechende Projektdatei zu öffnen und die projektabhängigen Daten darzustellen.

- Name und Pfad der Protokolldatei
- aktuelle Visualisierung (Sichtbarkeit der Attribute)
- Style
- Metrikparameter
- für jedes Fenster ein Fensterparametersatz, der aus folgenden Parametern besteht:
  - Größe
  - Position
  - Filter
  - Knotenpositionen
  - Zoom

Kapitel 3. GUI - Konzepte

Vorwort

Dieses Kapitel erläutert die Konzepte, die dem GUI von OpenCAGE zugrundeliegen. Da es in der Natur der GUI-Entwicklung liegt, dass ein GUI bis zur Auslieferung des fertigen Produktes mehreren Evaluierungen unterzogen wird, ist es auch erheblichen Änderungen unterworfen. Daher wird dieses Dokument während des gesamten Projektes an die jeweils aktuellen Änderungen angepasst.

Während des Projekts hilft dieses Dokument, die Konsistenz des GUI zu wahren und schafft eine Grundlage für das Benutzerhandbuch. Nach Beendigung des Projekts soll dieses Dokument dazu dienen, Orientierungshilfen für die weitere Entwicklung von OpenCAGE zu geben.

Dieses Kapitel ist sowohl Teil der Spezifikation als auch Teil einer eigenen Dokumenteneinheit. Daher kann es zu Redundanzen mit anderen Teilen der Spezifikation kommen (z.B. Programmkomponenten und GUI-Elemente).

Aufbau des GUI

Das GUI von OpenCAGE besteht im wesentlichen aus folgenden Elementen:

Workbench

Die *Workbench* (Abschnitt namens *Workbench* in Kapitel 4) dient verschiedenen Zwecken.

Zum einen bietet sie Zugang zu Funktionalität, die Einfluss auf die gesamte OpenCAGE-Sitzung hat, wie z.B. Laden und Speichern von RFGs.

Außerdem ist es von der Workbench aus mittels der Werkzeugleiste möglich, zwischen permanenten Modi hin- und herzuschalten, siehe *Werkzeugleiste / permanente Modi* (Abschnitt namens *Werkzeugleiste / permanente Modi*)

Außerdem kann der Benutzer mit Hilfe der Masterbar die *Master* (Abschnitt namens *Master*) öffnen, von denen aus u.a. temporäre Modi geschaltet werden können. Die Masterbar besteht aus einer Leiste von Buttons. Klickt man einen der Buttons an, so wird der dazugehörige Master geöffnet bzw. in den Vordergrund geholt, falls der Master schon geöffnet ist.

Master

Jeder Master (Abschnitt namens *Masters* in Kapitel 4) wird durch ein spezielles Fenster repräsentiert. Es kann pro Master immer nur ein Fenster geöffnet sein. Alle Master sind über die Masterbar erreichbar, siehe *Workbench* (Abschnitt namens *Workbench*).

Window-Master

Der Window-Master bietet eine schnelle Navigationsmöglichkeit über die Graph-Windows. Er kennt die Zuordnung von Graph-Windows zu Views und Knoten und die daraus resultierende hierarchische Abhängigkeit der Graph-Windows. Er realisiert gewohnte Funktionen auf Fenster wie z.B. Minimieren unter Ausnutzung der hierarchischen Struktur. Der Window-Master ist platzsparend und bietet daher all seine Funktionalität über Kontextmenüs an. Außerdem werden hier Fenster mit besonderer Bedeutung je nach Konfiguration, siehe *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* (Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* in Kapitel 2), farbig hervorgehoben, siehe *Zustände* (Abschnitt namens *Zustände*)

View-Master

Im View-Master werden alle Views dargestellt. Von hier aus können Operationen ausgeführt werden, die auf einer oder mehreren Views arbeiten. Views mit besonderer Bedeutung werden je nach Konfiguration, siehe *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* (Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* in Kapitel 2), grafisch hervorgehoben. Der View-Master ist eine besondere Art eines Graph-Windows.

Mode-Master

Mode-Master dienen dazu, Operationen auszuführen, die auf die Graph-Area wirken und für die der Benutzer in einem Dialog Informationen auswählen kann. Die Master sind daher zunächst aufgebaut wie *Dialoge* (Abschnitt namens *Dialoge*). Der Operationsname-Button gibt allerdings zusätzlich an, ob die Operation auf die gesamte Graph-Area oder nur auf einer Menge selektierter RFG-Objekte innerhalb einer Graph-Area ausgeführt wird, z.B. „Filter Window“. Dieser Button aktiviert einen temporären Modus.

Dieser temporäre Modus wird durch einen geänderten *Mauszeiger* (Abschnitt namens *Mauszeiger*) angezeigt. Der Benutzer klickt nun in die Graph-Area des gewünschten Graph-Windows, um die Aktion auszuführen. Damit ist der temporäre Modus wieder abgeschaltet.

Während der temporäre Modus aktiv ist, sind alle anderen Funktionen von OpenCAGE gesperrt. Um dies anzuzeigen, ändert sich der *Mauszeiger* (Abschnitt namens *Mauszeiger*), sobald der Benutzer den Mauszeiger über einem OpenCAGE-Fenster an eine Stelle führt, wo die ausgewählte Operation nicht ausgeführt werden kann. Klickt der Benutzer an einer solchen Stelle, dann wird der temporäre Modus damit abgeschaltet, ohne dass die Operation ausgeführt wurde.

Während der temporäre Modus aktiv ist, können alle anderen Programme außerhalb von OpenCAGE, und auch Funktionen des Betriebssystems, regulär ausgeführt werden.

Achtung! Beim Analyse-Master wählt der Benutzer ein Graph-Window aus, die Analyse wird jedoch auf der View ausgeführt, der das Graph-Window zugeordnet ist.

Graph-Window

In der Graph-Area der Graph-Windows werden die Graphen angezeigt, siehe *Visualisierung von Graphen* (Abschnitt namens *Visualisierung von Graphen*). Hier kann der Benutzer Operationen ausführen, die auf der gesamten Graph-Area bzw. auf selektierten RFG-Objekten arbeiten, z.B. Metriken anzeigen bzw. RFG-Objekte akzeptieren.

Direkt über der vertikalen Scrollbar befinden sich die Graph-Window-Buttons. Sie ermöglichen einen schnellen Zugang zu Operationen, die auf die Graph-Area ausgeführt werden, z.B. Selektieren aller RFG-Objekte eines Graph-Windows. Siehe auch *Selektieren* (Abschnitt namens *Selektieren*).

Um die Benutzung der Scrollbar nicht einzuschränken, sind die Buttons nur mit einem Buchstaben beschriftet. Es sollte nie mehr als drei Graph-Window-Buttons geben, siehe *Empty Graph Window* (Abschnitt namens *Empty Graph Window* in Kapitel 4).

Benutzeraktionen

Im folgenden werden die verschiedenen Möglichkeiten aufgelistet, Operationen und dazu erforderliche Daten auszuwählen und durchzuführen. Dabei werden zu den Interaktionsmöglichkeiten Richtlinien für Aufbau und Struktur der involvierten GUI-Komponenten (wie z.B. Menüs) aufgestellt.

Werkzeugleiste / permanente Modi

Die Werkzeugleiste besteht aus einer Reihe von Buttons, die das Verhalten von Schaltern nachbilden. Klickt der Benutzer einen Button an, dann bleibt dieser gedrückt bis der Benutzer einen anderen Button anklickt. Es kann immer nur ein Button gedrückt sein.

Jeder Button repräsentiert einen bestimmten Modus, der für die Graph-Area gilt. Entsprechend dem Modus verwandelt sich der Mauszeiger in ein anderes Symbol. Siehe auch *Mauszeiger* (Abschnitt namens *Mauszeiger*)).

Der Standard-Modus ist der Selektionsmodus. Nur in diesem Modus können Objekte selektiert und Operationen auf Objekte ausgeführt werden.

Selektieren und Markieren

Selektieren

Selektieren dient dazu, Objekte auszuwählen, auf denen Operationen ausgeführt werden können. Selektieren funktioniert im Window-Master, im View-Master, in einem Modus-Master, in den Graph-Windows und in einem Select-Dialog. Selektierte Objekte werden grafisch hervorgehoben, siehe *Zustände* (Abschnitt namens *Zustände*). Das Selektieren ist nicht zu verwechseln mit dem *Markieren* (Abschnitt namens *Markieren*), welches lediglich dazu dient, dem Benutzer visuelles Feedback zu geben und das nur in Graph-Windows zur Verfügung steht.

Window-Master, View-Master und Graph-Windows stellen drei getrennte Einheiten dar, d.h. das Selektieren in einer der drei Einheiten beeinflusst Selektionen in den jeweils anderen Einheiten nicht.

Der Benutzer selektiert ein einzelnes Element, indem er es mit der linken Maustaste anklickt. Damit werden alle bisher selektierten Objekte dieser Einheit automatisch deselektiert.

Ein einzelnes Element kann außerdem selektiert werden, indem es der Benutzer mit der rechten Maustaste anklickt. Damit wird zugleich ein Kontextmenü (siehe *Kontextmenüs* (Abschnitt namens *Kontextmenüs*)) aufgerufen, aus dem eine Operation ausgewählt werden kann, die unmittelbar auf das selektierte Element wirkt.

Im View-Master sowie in Graph-Windows ist es möglich, mehrere Objekte zu selektieren. Dies geht zum einen, indem der Benutzer die **Shift**-Taste drückt, während er mit der linken Maustaste die gewünschten Objekte nacheinander anklickt. Zum anderen ist es möglich, einen Rahmen um Objekte zu ziehen, die selektiert werden sollen, indem der Benutzer mit der linken Maustaste auf den Hintergrund einer Graph-Area klickt, die Maustaste gedrückt hält und damit einen Rahmen zieht, siehe *Select and Mark* (Abschnitt namens *Select and Mark* in Kapitel 5). Darüberhinaus ist es möglich, durch Ziehen eines Rahmen bei gedrückter **Shift**-Taste mehrere Objekte zu einer Selektion hinzuzufügen.

Klickt der Benutzer mit der rechten Maustaste auf eines der selektierten Objekte, so bleibt diese Selektion erhalten und die Operation, die aus dem Kontextmenü ausgewählt wird, wirkt auf alle selektierten Objekte.

Im View-Master und in Graph-Windows können Objekte deselektiert werden, indem auf eine Stelle der Graph-Area geklickt wird, an der sich kein Objekt befindet. *Beachte:* Der View-Master ist ein besonderes Graph-Window.

In Graph-Windows gibt es bei den Graph-Window-Buttons oberhalb der vertikalen Scroll-Bar einen Select-All-Button. Dieser selektiert alle RFG-Objekte im Graph-Window.

Mit Hilfe des Invert-Selection-Buttons kann die Selektion von RFG-Objekten im Graph-Window umkehrt werden.

Markieren

Das Markieren ist eine Operation auf selektierte RFG-Objekte. Zum Markieren und Demarkieren stehen daher alle Kombinationsmöglichkeiten zur Verfügung, die das Selektieren bietet. Abgesehen davon beeinflussen sich Markieren und Selektieren gegenseitig nicht.

Will der Benutzer RFG-Objekte markieren, dann selektiert er diese und führt die Operation *Mark RFG-Object by Selection* (Abschnitt namens *Mark RFG-Object by Selection* in Kapitel 5) aus. Es können auf diese Art auch beliebig viele RFG-Objekte zusätzlich markiert werden, ohne die bestehenden Markierungen zu beeinflussen.

Will der Benutzer RFG-Objekte demarkieren, dann selektiert er diese und führt die Operation *Unmark RFG-Object by Selection* (Abschnitt namens *Unmark RFG-Object by Selection* in Kapitel 5) aus.

Will der Benutzer alle RFG-Objekte demarkieren, dann führt er die Operation *Unmark All RFG-Objects* (Abschnitt namens *Unmark All RFG-Objects* in Kapitel 5) aus.

Menüs

Menüs haben nicht mehr als drei Ebenen. Sämtliche Menüstrukturen werden in Dateien gespeichert, so dass sie von den Entwicklern ohne größeren Aufwand verändert werden können.

Menü-Einträge sind ausgegraut, wenn die Vorbedingungen der dazugehörigen Operationen nicht erfüllt sind (z.B. Copy, wenn kein Knoten selektiert ist). Ansonsten ändern sich Menü-Einträge während einer OpenCAGE-Sitzung nicht.

Gibt es eine Operation, die auf verschiedenen Objekttypen (z.B. Views und selektierte Knotenmengen) arbeiten kann, dann gelten folgende Regeln:

Tabelle 3-1. Regeln für den Menüaufbau

Bedingung	Visualisierung
Die Operation ist in Ebene 1 oder 2 der Menühierarchie und es gibt viele verschiedene Objekttypen, auf die die Operation wirken kann.	Es gibt einen Menüpunkt **Operationsname**, der Submenüs Objekttyp1, Objekttyp2, ... enthält.
Die Operation kann auf maximal drei verschiedene Objekttypen wirken und/oder die Operation befindet sich in Ebene drei der Menühierarchie.	Es gibt Menüpunkte **Operationsname Objekttyp1**, **Operationsname Objekttyp2**, ... in derselben Ebene.
Zusätzlich zu den Objekttypen muss eine Reihe weiterer Informationen bestimmt werden, um die Operation durchzuführen. Diese Informationen werden für andere Operationen ebenfalls benötigt.	Es existiert ein Menüpunkt, der einen Dialog öffnet, in dem der Benutzer den Objekttyp auswählt, auf den die Operation ausgeführt wird. Weitere Operationen greifen ebenfalls auf diesen Dialog zu.

Kontextmenüs

Für jede der GUI-Einheiten Window-Master, View-Master und Graph-Window gibt es ein eigenes Kontextmenü. Das Kontextmenü wird angezeigt, wenn der Benutzer mit der rechten Maustaste auf eine der GUI-Einheiten klickt. Es werden nur solche Operationen in einem Kontextmenü angeboten, die für die entsprechende GUI-Einheit verwendet werden können.

Dialoge

Buttons

Folgende Regeln gelten bezüglich Buttons in Dialogen:

- Viele Dialoge besitzen einen **Help**-Button. Klickt der Benutzer diesen Button an, dann wird die Hilfeseite zum entsprechenden Dialog im Browser angezeigt. Ist noch kein Browser offen, dann startet OpenCAGE den vom Benutzer angegebenen Browser (siehe *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* (Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* in Kapitel 2)). Dem Browser wird die entsprechende URL übergeben. Das Hilfesystem wird im Rahmen dieses Projekts nur vorgesehen, aber nicht implementiert.
- Buttons, die sich auf den ganzen Dialog beziehen, sind am unteren Rand angeordnet. Es sind nicht in jedem Dialog alle Buttons vorhanden. Von links nach rechts haben sie folgende Reihenfolge:
  - Operationsname / OK
  - Cancel
  - Close
  - Help
- Buttons, die sich nur auf einen Teil vom Dialog beziehen, üblicherweise auf die linke oder die rechte Hälfte, werden zu diesem Dialogteil gruppiert.

- Für die Vorselektion gelten folgende Regeln:
  - Handelt es sich um eine Warnung oder um einen Dialog für eine destruktive Aktion (wie z.B. Löschaktionen), so ist der **Cancel**-Button selektiert.
  - Handelt es sich um einen Mode-Master, so ist der Button, der sich am weitesten links befindet, selektiert.
  - Bei allen anderen Dialogen ist der **OK**-Button selektiert.

- Ausnahmen bilden Dialoge, die nicht in OpenCAGE implementiert sind, sondern nur von OpenCAGE aus aufgerufen werden, wie z.B. der Dialog zum Auswählen von Dateien.

Parameter

Dialoge, die zum Bearbeiten von Parametereinstellung dienen, werden vertikal in zwei Hälften geteilt. In der linken Hälfte wird ein Baum bzw. eine Liste mit Elementen angezeigt. In der rechten Hälfte werden die Parameter zum links ausgewählten Element angezeigt und können bearbeitet werden.

Passen die Parameter nicht auf eine Dialogseite, dann werden sie zu Gruppen zusammengefasst und in einem Notizbuch dargestellt.

Texteingabefelder

Dialoge, die nur ein einzelnes Texteingabefeld enthalten, haben auf der linken Seite ein Fragezeichen-Symbol.

Prinzipiell ist es erlaubt, Zeichen aus dem gesamten Latin1-Zeichensatz einzugeben. Dies wird beschränkt durch die RFG-Bibliothek und GtkAda.

Shortcuts

Shortcuts bieten durch Drücken von Tastenkombinationen einen schnelleren Zugang zu Operationen, die der Benutzer oft ausführt. Auch die Shortcuts können vom Benutzer in einer Datei frei konfiguriert werden, siehe *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* (Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* in Kapitel 2). Dabei sind alle Buchstaben- und Zahlentasten in Kombination mit den Tasten **Strg** bzw. **Ctrl** und **Alt** erlaubt.

Gibt es mehrere Zuordnungen für eine Tastenkombination, dann wird die jeweils letzte verwendet.

Achtung! Der Benutzer hat selber Sorge zu tragen, dass es keine Doppelbelegungen von Tastenkombinationen durch das Betriebssystem und OpenCAGE gibt.

Feedback

Aktionen

Andauernde Operationen

OpenCAGE berechnet vor bestimmten Operationen (wie Layout-Operationen) deren zu erwartende Dauer. Braucht eine Operation voraussichtlich länger als eine Sekunde, dann wird anstatt des Standard-Pointers der Busy-Pointer angezeigt, solange sich der *Mauszeiger* (Abschnitt namens *Mauszeiger*) über einem Fenster befindet, das von der ausgeführten Operation betroffen ist.

Startet der Benutzer eine Analyse, dann erscheint ein Fenster, in dem das Fortschreiten der Analyse dargestellt wird.

Mauszeiger

Normalerweise wird der Standard-Pointer als Mauszeiger angezeigt. Unter bestimmten Bedingungen kommt es aber vor, dass der Mauszeiger durch ein anderes Symbol dargestellt wird, siehe *Pointer* (Abschnitt namens *Pointer* in Kapitel 4).

Zoom-Pointer

: Der Zoom-Pointer wird angezeigt, wenn der Benutzer in der Werkzeugleiste in den Zoom-Mode umgeschaltet hat.

Busy-Pointer

: Der Busy-Pointer wird angezeigt, wenn eine Operation voraussichtlich länger als eine Sekunde dauert.

Paste-Pointer

: Hat der Benutzer die Paste-Operation ausgewählt, dann wird der Mauszeiger als Paste-Pointer dargestellt, solange er sich über einer Stelle befindet, auf die die Paste-Operation ausgeführt werden darf.

No-Pointer

: Wenn der Benutzer gerade im Begriff ist, eine Operation auszuführen, für die er in einen anderen Modus als den Standard-Modus (Selection-Mode) schalten musste, dann verwandelt sich der Mauszeiger immer dann in den No-Pointer, wenn er sich gerade über einer Stelle befindet, auf der die gewählte Operation nicht ausgeführt werden darf.

Außerdem gibt es für die temporären Modi der *Mode-Master* (Abschnitt namens *Mode-Master*) folgende Pointer:

- Analyze-Window-Pointer
- Filter-Selection-Pointer
- Filter-Window-Pointer
- Layout-Selection-Pointer
- Layout-Window-Pointer
- Metrics-Window-Pointer

- Move-Pointer
- No-Parking-Pointer
- Unfilter-Pointer

Zustände

Read-Only-View

In allen Graph-Windows, die auf eine Read-Only-View geöffnet sind, wird links unten neben der horizontalen Scrollbar ein farbig unterlegtes `R` angezeigt.

aktuelle User-View

In allen Graph-Windows, die auf die aktuelle User-View geöffnet sind, wird links unten neben der horizontalen Scrollbar ein farbig unterlegtes `U` angezeigt.

Aktiver Filter

In allen Graph-Windows, in denen ein Filter gesetzt ist, wird links unten neben der horizontalen Scrollbar ein farbig unterlegtes `F` angezeigt.

RFG-Objekt-Highlighting

Es gibt für RFG-Objekte Select-Highlighting, Mark-Highlighting und Conflict-Highlighting.

Select-Highlighting dient dazu, die selektierten Objekte hervorzuheben, auf die eine Operation ausgeführt wird. Selektierte RFG-Objekte werden in allen Graph-Windows hervorgehoben, die derselben View zugeordnet sind, wie das Graph-Window, in dem selektiert wurde. Select-Highlighting gibt es auch im Window-Master und im View-Master. Hier wird der selektierte Fenstername mit einem Balken hinterlegt bzw. die selektierte(n) View(s) eingefärbt.

Mark-Highlighting dient dazu, gleiche RFG-Objekte in verschiedenen Views zu visualisieren.

Conflict-Highlighting dient dazu, diejenigen RFG-Objekte grafisch hervorzuheben, die bei einer Kopier- oder Verschiebeoperation zu Konflikten geführt haben.

Visualisierung von Graphen

Knoten

Unterschiedliche Knotentypen werden durch unterschiedliche Pixmaps dargestellt. Für jeden Stil gibt es fünf verschiedene Pixmaps:

- Normal
- Selection-Highlighting
- Mark-Highlighting
- Conflict-Highlighting

- Kontur

Der Benutzer kann die Pixmaps beliebig austauschen, siehe *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* (Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* in Kapitel 2).

Knotennamen werden direkt unterhalb des Knotens angezeigt.Der Knotenname kann aus beliebigen Bestandteilen der unten aufgeführten Teile bestehen. Dies wird in den *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* (Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* in Kapitel 2) festgelegt. Die wirkliche Darstellung des Knotenamens kann für Entitätsknoten und Composite-Knoten unterschiedlich sein. Knotennamen bestehen maximal aus den Teilen:

- Projektname
- Pfad
- Dateiname
- Modulname
- Objektname

Kanten

Jeder Kantentyp wird durch einen bestimmten Kantenstil visualisiert, wie z.B. Linienart oder Liniendicke. Wie bei den *Knoten* sind auch hier fünf verschiedene Stile definiert.

Überlappende Kanten

Folgende Regeln gelten im Falle von Multikanten: Bis zu drei Kanten werden voneinander getrennt dargestellt. Ab vier Kanten werden die Multikanten durch eine dicke Kante ersetzt. Bleibt man mit dem Mauszeiger auf einer solchen Kante eine Sekunde stehen, dann werden im Tooltip die Namen der Kanten und Kantentypen angezeigt. Alternativ dazu kann der Benutzer auch durch einen Doppelklick mit der linken Maustaste den Namen der Kanten und Kantentypen anzeigen lassen.

Liegen mehrere Knoten in einer Linie und sind diese Knoten in einer solchen Weise durch Kanten verbunden, dass sich die Kanten überlappen würden, dann sollten die Kanten im Bogen um die Knoten geführt werden, so dass es zu keiner Überlappung kommt.

Unvollständige Kanten

Unvollständige Kanten gehen vom dargestellten Knoten zum dem am nächsten liegenden Rahmen des Graph-Windows.

Kapitel 4. GUI - Realisierung

Allgemeines

Workbench

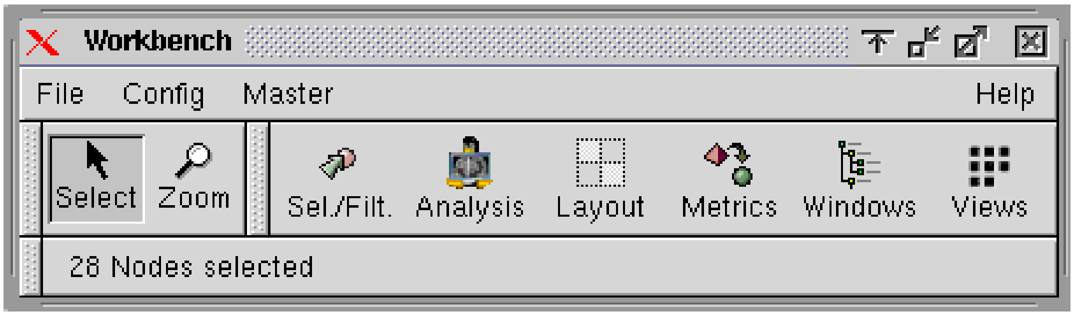


Abbildung 4-1. Dialog: Workbench

Die Kommandozentrale; hier können allgemeine Einstellungen gemacht und Aktionen ausgelöst werden. Die Workbench ist jederzeit im Vordergrund und kann mit der Maus an jede Position auf dem Bildschirm gezogen werden. Die Workbench besteht aus einer Menüzeile, einer Masterleiste, einer Werkzeugleiste und einer Statusleiste. Die Workbench erscheint direkt nach dem Programmstart. Die Workbench ist während des ganzen Programmablaufs geöffnet, kann aber auch minimiert werden.

Menüleiste

File Menu

File⟶**Open...**: *Open RFG* (Abschnitt namens *Open RFG* in Kapitel 5)

File⟶**Close**: *Close RFG* (Abschnitt namens *Close RFG* in Kapitel 5)

File⟶**Save**: *Save RFG* (Abschnitt namens *Save RFG* in Kapitel 5)

File⟶**Save as...**: *Save RFG as* (Abschnitt namens *Save RFG as* in Kapitel 5)

File⟶**Script...**: *Run Script* (Abschnitt namens *Run Script* in Kapitel 5)

File⟶**Exit**: *Exit* (Abschnitt namens *Exit* in Kapitel 5)

Edit Menu

Edit⟶**Undelete**: *Undelete* (Abschnitt namens *Undelete* in Kapitel 5)

Edit⟶**Clear Trashcan**: *Clear Trashcan* (Abschnitt namens *Clear Trashcan* in Kapitel 5)

Config Menu

Config⟶**Edit...**: *Edit Config* (Abschnitt namens *Edit Config* in Kapitel 5)

Config⟶**Reload**: *Reload Config* (Abschnitt namens *Reload Config* in Kapitel 5)

Config⟶Affinity Threshold...: *Modify Affinity Threshold* (Abschnitt namens *Modify Affinity Threshold* in Kapitel 5)

Master Menu

Master⟶Filter-and-Selection...: *Filter to Selection* (Abschnitt namens *Filter to Selection* in Kapitel 5), *Filter to Window* (Abschnitt namens *Filter to Window* in Kapitel 5), *Unfilter* (Abschnitt namens *Unfilter* in Kapitel 5)oder *Select RFG-Objects by Criteria* (Abschnitt namens *Select RFG-Objects by Criteria* in Kapitel 5)

Master⟶Analysis...: *Analysis* (Abschnitt namens *Analysis* in Kapitel 5)

Master⟶Layout...: *Assign Layout to Selection* (Abschnitt namens *Assign Layout to Selection* in Kapitel 5) oder *Assign Layout to Window* (Abschnitt namens *Assign Layout to Window* in Kapitel 5)

Master⟶Metrics...: *Show Metric* (Abschnitt namens *Show Metric* in Kapitel 5)

Master⟶Windows...: *Window-Master* (Abschnitt namens *Window-Master* in Kapitel 5)

Master⟶Views...: *View-Master* (Abschnitt namens *View-Master* in Kapitel 5)

Help Menu

Help⟶About...: *About-Dialog* (Abschnitt namens *About-Dialog* in Kapitel 5)

Help⟶Context Help...: *Context Help* (Abschnitt namens *Context Help* in Kapitel 5)

Help⟶Help Index...: *Help Index* (Abschnitt namens *Help Index* in Kapitel 5)

Werkzeugleiste

In der Werkzeugleiste wird ausgewählt, welche Aktion ein Klick mit der Maus in einem Graph-Window ausführt. Nach dem Start von OpenCAGE ist als Werkzeug „Select“ ausgewählt.

Folgende Werkzeuge sind auswählbar:

- Select ⇒ Ein Klick mit der linken Maustaste in einem Graph-Window führt einen der Use-Cases *Select Single RFG-Object by Mouse* (Abschnitt namens *Select Single RFG-Object by Mouse* in Kapitel 5), *Select Multiple RFG-Objects by Mouse* (Abschnitt namens *Select Multiple RFG-Objects by Mouse* in Kapitel 5), *Move Nodes* (Abschnitt namens *Move Nodes* in Kapitel 5), *Add / Remove Single RFG-Object by Mouse to / from Selection* (Abschnitt namens *Add / Remove Single RFG-Object by Mouse to / from Selection* in Kapitel 5), *Add Multiple RFG-Objects by Mouse to Selection* (Abschnitt namens *Add Multiple RFG-Objects by Mouse to Selection* in Kapitel 5) aus.

  Ein Klick mit der rechten Maustaste in einem Graph-Window öffnet ein Kontextmenü.

- Zoom ⇒ Ein Klick mit der linken Maustaste in einem Graph-Window führt den Use-Case *Zoom In* (Abschnitt namens *Zoom In* in Kapitel 5) aus; ein Klick mit der rechten Maustaste in einem Graph-Window führt den Use-Case *Zoom Out* (Abschnitt namens *Zoom Out* in Kapitel 5) aus.

Masterleiste

In der Masterleiste können verschiedene Master geöffnet bzw. in den Vordergrund geholt werden. Durch einen Klick mit der linken Maustaste wird der Master geöffnet. Ist der betreffende Master bereits geöffnet, so wird dieser in den Vordergrund geholt.

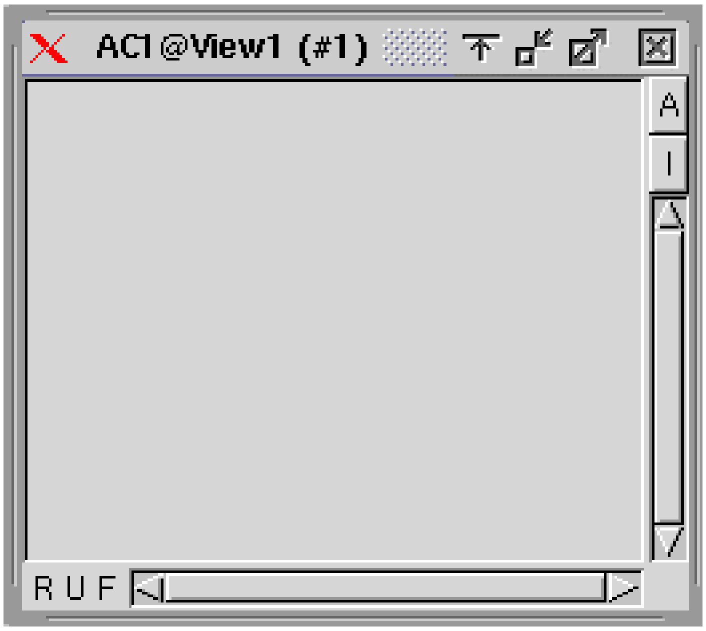
Folgende Masters sind auswählbar:

- Filter-and-Select-Master: *Filter to Window* (Abschnitt namens *Filter to Window* in Kapitel 5), *Filter to Selection* (Abschnitt namens *Filter to Selection* in Kapitel 5), *Unfilter* (Abschnitt namens *Unfilter* in Kapitel 5) oder *Select RFG-Objects by Criteria* (Abschnitt namens *Select RFG-Objects by Criteria* in Kapitel 5)
- Analysis-Master: *Analysis* (Abschnitt namens *Analysis* in Kapitel 5)
- Layout-Master: *Assign Layout to Selection* (Abschnitt namens *Assign Layout to Selection* in Kapitel 5) oder *Assign Layout to Window* (Abschnitt namens *Assign Layout to Window* in Kapitel 5)
- Metric-Master: *Show Metric* (Abschnitt namens *Show Metric* in Kapitel 5) oder *Hide Metric* (Abschnitt namens *Hide Metric* in Kapitel 5)
- Window-Master: *Window-Master* (Abschnitt namens *Window-Master* in Kapitel 5)
- View-Master: *View-Master* (Abschnitt namens *View-Master* in Kapitel 5)

Statusleiste

In der Statusleiste wird die Anzahl der selektierten Knoten angezeigt.

Empty Graph Window



In diesem Bild sind die in jedem Graph-Fenster vorhandenen Buttons über der rechten Scrollbar und der Statusbereich links neben der unteren Scrollbar zu sehen.

Abbildung 4-2. Dialog: Empty Graph Window

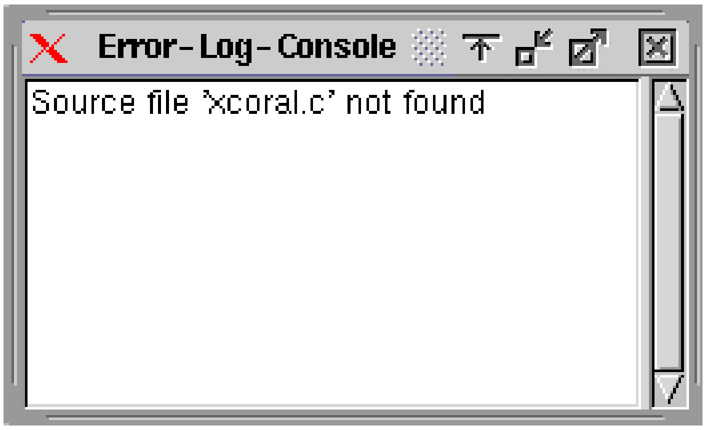
- Der **A**-Button selektiert alle RFG-Objekte, die in dem betreffenden Graph-Window liegen.
- Der **I**-Button selektiert alle RFG-Objekte, die in dem betreffenden Graph-Window liegen und nicht selektiert sind und deslektiert alle RFG-Objekte, die in dem betreffenden Graph-Window liegen und selektiert sind.
- Im Statusbereich werden verschiedene Zustände des Fensters bzw. der View signalisiert.

  Ein sichtbares `R` erscheint genau dann, wenn die View Read-Only gesetzt ist.

  Ein sichtbares `U` erscheint genau dann, wenn die View die aktuelle Userview ist.

  Ein sichtbares `F` erscheint genau dann, wenn die View gefiltert angezeigt ist.

Error-Log-Console



In diesem Fenster werden alle eventuell auftretenden Fehler protokolliert, die das Betriebssystem, die benutzten Bibliotheken oder OpenCAGE selbst melden.

Abbildung 4-3. Dialog: Error-Log-Console

Kontextmenü

Im folgenden sind die einzelnen Fenster und ihre Kontextmenüs beschrieben. Die Funktionen, die innerhalb der Kontextmenüs aufgerufen werden können, sind bei den entsprechenden Abschnitten im Kapitel *Use Cases* (Kapitel 5) beschrieben. Wegen den Vorbedingungen zu den einzelnen Funktionen sei auf die entsprechende Use-Case-Beschreibung verwiesen.

Window-Master

Der Window-Master besitzt ein Kontextmenü, das bei einem Rechtsklick auf ein beliebiges Element innerhalb des Fensters des Window-Masters erscheint. Das Kontextmenü enthält folgende Einträge:

- Close
  - auf Fenstereinträgen: *Close Window* (Abschnitt namens *Close Window* in Kapitel 5)
  - auf Grapheneinträgen: *Close Windows Hierarchically* (Abschnitt namens *Close Windows Hierarchically* in Kapitel 5)
- Minimize
  - auf Fenstereinträgen: *Minimize Window* (Abschnitt namens *Minimize Window* in Kapitel 5)
  - auf Grapheneinträgen: *Minimize Windows Hierarchically* (Abschnitt namens *Minimize Windows Hierarchically* in Kapitel 5)
- Restore

  - auf Fenstereinträgen: *Restore Window* (Abschnitt namens *Restore Window* in Kapitel 5)
  - auf Grapheneinträgen: *Restore Windows Hierarchically* (Abschnitt namens *Restore Windows Hierarchically* in Kapitel 5)
- Show Window: *Open Window For Displayed Graph Component* (Abschnitt namens *Open Window For Displayed Graph Component* in Kapitel 5)

Graph-Window

Ein Graph-Window hat, wenn es dargestellt wird, ein Kontextmenü mit folgenden Einträgen:

- RFG

  RFG⟶Accept: *Accept* (Abschnitt namens *Accept* in Kapitel 5)

  RFG⟶Accept as: *Accept as* (Abschnitt namens *Accept as* in Kapitel 5)

  RFG⟶Reject: *Reject* (Abschnitt namens *Reject* in Kapitel 5)

  RFG⟶Collapse: *Collapse* (Abschnitt namens *Collapse* in Kapitel 5)

  RFG⟶Expand: *Expand* (Abschnitt namens *Expand* in Kapitel 5)

- Edit

  Edit⟶Copy: *Copy* (Abschnitt namens *Copy* in Kapitel 5)

  Edit⟶Cut: *Cut* (Abschnitt namens *Cut* in Kapitel 5)

  Edit⟶Paste: *Paste* (Abschnitt namens *Paste* in Kapitel 5)

- Attributes

  Attributes⟶Show: *Show attributes* (Abschnitt namens *Show attributes* in Kapitel 5)

  Attributes⟶Hide: *Hide attributes* (Abschnitt namens *Hide attributes* in Kapitel 5)

  Attributes⟶Edit: *Edit Attributes* (Abschnitt namens *Edit Attributes* in Kapitel 5)

- Conflict-Highl

  Conflict-Highl⟶Remove by Sel: *Remove Conflict-Highlighting to Selection* (Abschnitt namens *Remove Conflict-Highlighting to Selection* in Kapitel 5)

  Conflict-Highl⟶Remove all: *Remove Conflict-Highlighting Everywhere* (Abschnitt namens *Remove Conflict-Highlighting Everywhere* in Kapitel 5)

- **Annotate**: *Edit Node Annotation* (Abschnitt namens *Edit Node Annotation* in Kapitel 5)
- others

  others⟶Rename: *Rename Node* (Abschnitt namens *Rename Node* in Kapitel 5)

others⟶Show/Hide: *Show/Hide edges of other views* (Abschnitt namens *Show/Hide edges of other views* in Kapitel 5)

View-Master

Der View-Master enthät ein Kontextmenü mit folgenden Einträgen:

- Change read-only: *Change read-only* (Abschnitt namens *Change read-only* in Kapitel 5)
- User-View

  User-View⟶Set as current: *Set as current User-View* (Abschnitt namens *Set as current User-View* in Kapitel 5) **User-View⟶Create**: *Create new User-View* (Abschnitt namens *Create new User-View* in Kapitel 5) **User-View⟶Delete**: *Delete User-View* (Abschnitt namens *Delete User-View* in Kapitel 5)

- Make

  Make⟶Set Union: *Set Union* (Abschnitt namens *Set Union* in Kapitel 5)

  Make⟶Intersect: *Intersection* (Abschnitt namens *Intersection* in Kapitel 5)

  Make⟶Difference: *Difference* (Abschnitt namens *Difference* in Kapitel 5)

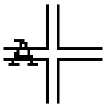
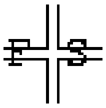
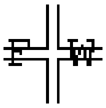
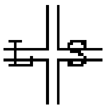
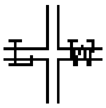
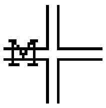
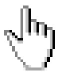
- **Annotate**: *Edit View Annotation* (Abschnitt namens *Edit View Annotation* in Kapitel 5)

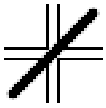
Pointer

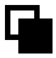
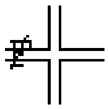
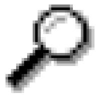
In der folgenden Tabelle sind alle Pointer aufgeführt, die in OpenCAGE verwendet werden. Unterschiedliche Pointer verdeutlichen unterschiedliche Modi, die OpenCAGE einnehmen kann. Dabei wird zwischen permanenten und temporären Modi unterschieden. Die folgende Tabelle gibt Auskunft, welche Pointer für welche Zwecke verwendet werden. Der Standard-Pointer und der Busy-Pointer nehmen dabei eine Sonderstellung ein. Sie kommen in allen Use-Cases zum Einsatz. Ist nichts anderes festgelegt, wird davon ausgegangen, dass sich OpenCAGE im Standard-Modus befindet, vergleiche *Mauszeiger*, (Abschnitt namens *Mauszeiger* in Kapitel 3).

Tabelle 4-1. Pointer-Übersicht

Symbol	Name	Modus	Use-Case

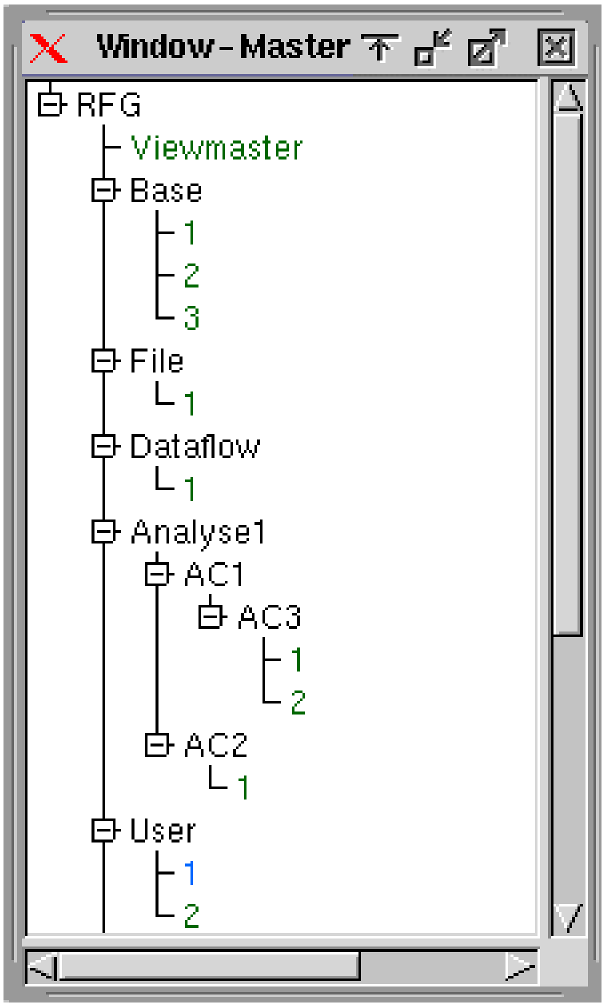
Symbol	**Name**	**Modus**	**Use-Case**
systemabhängig, z.B. 	Standard-Pointer	Standard-Modus	überall
systemabhängig	Busy-Pointer	während längeren Berechnungen	überall
	Analyze-Window-Pointer	während der Analyze-Aktion	• *Analysis* (Abschnitt namens *Analysis* in Kapitel 5)
	Filter-Selection-Pointer	während der Filter-to-Selection-Aktion	• *Filter to Selection* (Abschnitt namens *Filter to Selection* in Kapitel 5)
	Filter-Window-Pointer	während der Filter-to-Window-Aktion	• *Filter to Window* (Abschnitt namens *Filter to Window* in Kapitel 5)
	Layout-Selection-Pointer	während der Assign-Layout-to-Selection-Aktion	• *Assign Layout to Selection* (Abschnitt namens *Assign Layout to Selection* in Kapitel 5)
	Layout-Window-Pointer	während der Assign-Layout-to-Window-Aktion	• *Assign Layout to Window* (Abschnitt namens *Assign Layout to Window* in Kapitel 5)
	Metrics-Window-Pointer	während der Show-Metric- oder Hide-Metric-Aktion	• *Hide Metric* (Abschnitt namens *Hide Metric* in Kapitel 5) • *Show Metric* (Abschnitt namens *Show Metric* in Kapitel 5)
	Move-Pointer	während der Move-Nodes-Aktion	• *Move Nodes* (Abschnitt namens *Move Nodes* in Kapitel 5)

Symbol	**Name**	**Modus**	**Use-Case**
	No-Pointer	nach Aktionsklick im Master in einem Window von OpenCAGE, jedoch außerhalb der Graph-Area	• *Select RFG-Objects by Criteria* (Abschnitt namens *Select RFG-Objects by Criteria* in Kapitel 5) • *Assign Layout to Selection* (Abschnitt namens *Assign Layout to Selection* in Kapitel 5) • *Assign Layout to Window* (Abschnitt namens *Assign Layout to Window* in Kapitel 5) • *Filter to Selection* (Abschnitt namens *Filter to Selection* in Kapitel 5) • *Filter to Window* (Abschnitt namens *Filter to Window* in Kapitel 5) • *Unfilter* (Abschnitt namens *Unfilter* in Kapitel 5) • *Hide Metric* (Abschnitt namens *Hide Metric* in Kapitel 5) • *Show Metric* (Abschnitt namens *Show Metric* in Kapitel 5) • *Analysis* (Abschnitt namens *Analysis* in Kapitel 5)

Symbol	**Name**	**Modus**	**Use-Case**
	No-Parking-Pointer	RFG-Object außerhalb der Graph-Area des aktuellen Graph-Windows bei einer Verschieb-Aktion	• *Move Nodes* Abschnitt namens *Move Nodes* in Kapitel 5
	Paste-Pointer	während der Paste-Aktion	• *Paste* (Abschnitt namens *Paste* in Kapitel 5)
	Unfilter-Pointer	während der Unfilter-Aktion	• *Unfilter* (Abschnitt namens *Unfilter* in Kapitel 5)
	Zoom-Pointer	Zoom-Modus	• *Zoom In* (Abschnitt namens *Zoom In* in Kapitel 5) • *Zoom Out* (Abschnitt namens *Zoom Out* in Kapitel 5)

Masters

Window-Master

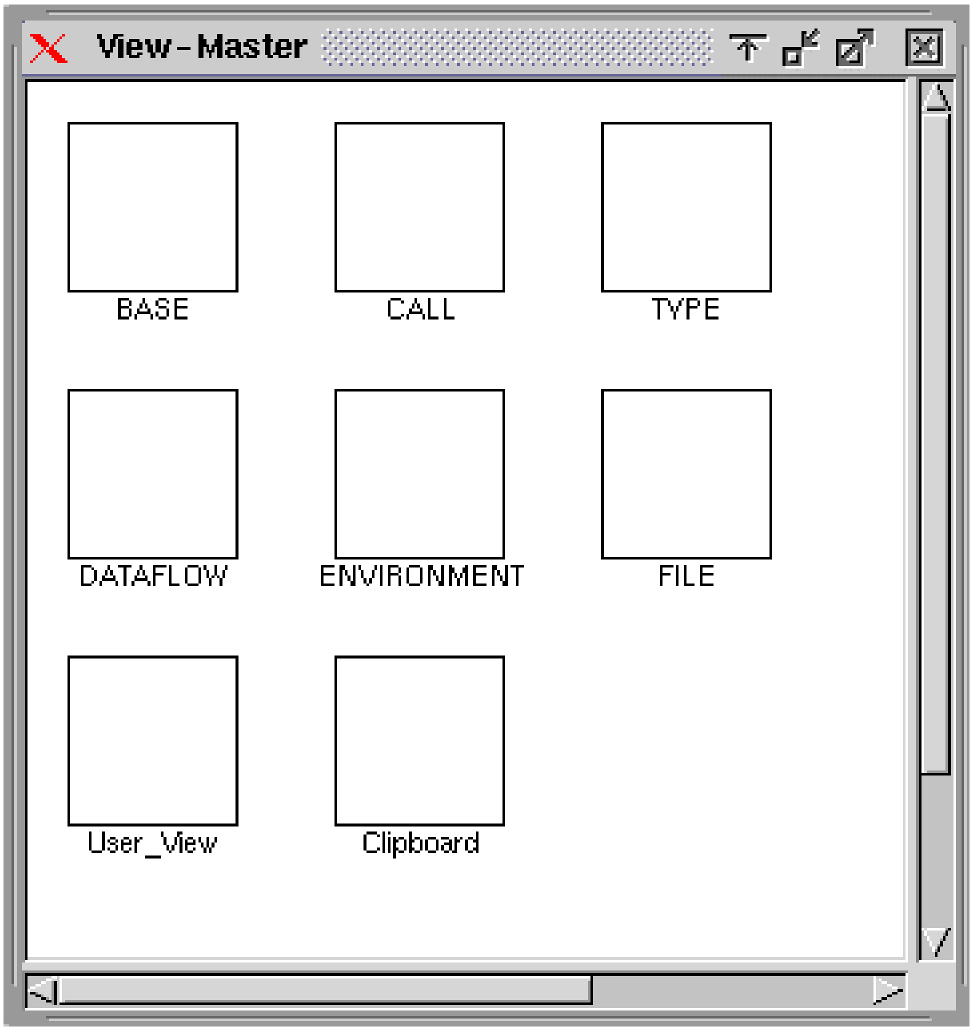


In diesem Dialog können die Graph-Windows verwaltet werden.

Abbildung 4-4. Dialog: Window-Master

- Grapheneinträge: Die Einträge erscheinen in einer in der Konfigurationsdatei (Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* in Kapitel 2) auszuwählenden Farbe.
- Fenstereinträge: Die Einträge erscheinen in einer in der Konfigurationsdatei (Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* in Kapitel 2) auszuwählenden Farbe.

View-Master



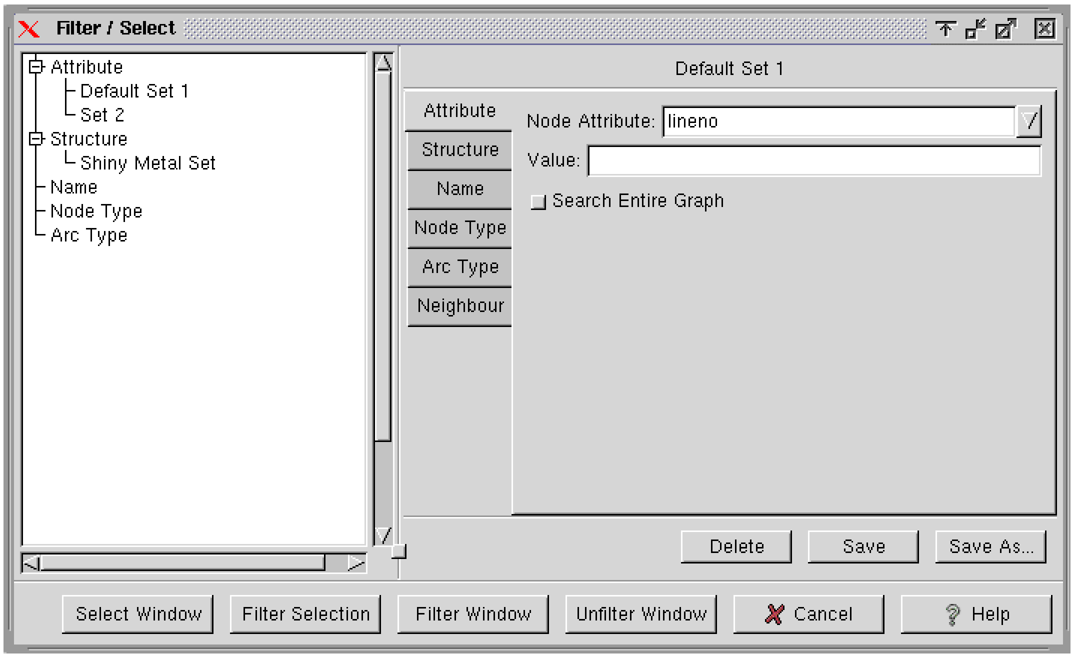
In diesem Dialog können die Views verwaltet werden.

Abbildung 4-5. Dialog: View-Master

- Im View-Master sind alle Views enthalten, die im aktuellen Projekt vorhanden sind. Der View-Master ist ein Graph Window, in dem die Views als Knoten wie in einem Graph-Window dargestellt werden, d.h es lassen sich alle Operationen, die in einem Graph-Window möglich sind, auch im View-Master ausführen
- Wenn eine View read-only ist, wird dies durch eine entsprechende Pixmap visualisiert, siehe *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* (Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* in Kapitel 2).

Mode-Master

Filter/Select-Master



In diesem Master können Filter- und Selektionsparameter eingestellt werden. Auch Filter- und Selektionsaktionen werden hier ausgelöst.

Abbildung 4-6. Dialog: Filter/Select-Master

- Im linken Teil des Dialogs befindet sich eine Liste der bisher abgespeicherten Parametersätze. Durch Anklicken eines Parametersatzes werden die entsprechenden Kriterien auf der rechten Seite dargestellt.
- Im rechten Teil befindet sich ein Notizbuch, über dessen Reiter die Art der Kriterien ausgewählt wird, nach denen selektiert wird.

  Es werden immer nur die Kriterien berücksichtigt, die sich auf dem gerade sichtbaren Blatt befinden. Eine Kombination der Kriterien mehrerer Blätter ist nicht möglich.

- Auf dem jeweiligen Notizbuch-Blatt sind dann die Kriterien sichtbar.

  Attribute

  > Es gibt eine Drop-Down-Liste, in der man eines der Knotenattribute auswählen kann. Man kann den Wert, den das Attribut haben soll, angeben. Außerdem gibt es eine Checkbox „Search Entire Graph“. Ist diese ausgewählt, wird der gesamte Graph durchsucht, ansonsten nur das später zu wählende Graph-Window.

Structure (analog zu Rigi)

Es gibt eine Liste, in der man beliebig viele Knotentypen auswählen kann. Zu dieser gibt es folgende Buttons:

- **All**-Button, mit dem alle Knotentypen ausgewählt werden können.
- **Clear**-Button, mit dem alle Knotentypen deselektiert werden Können.
- **Complement**-Button, mit dem die Auswahl invertiert werden kann.

Außerdem existiert eine Liste, in der man beliebig viele Kantentypen auswählen kann. Zu dieser gibt es folgende Buttons:

- **All**-Button, mit dem alle Kantentypen ausgewählt werden können.
- **Clear**-Button, mit dem alle Kantentypen deselektiert werden Können.
- **Complement**-Button, mit dem die Auswahl invertiert werden kann.

Außerdem gibt es eine Drop-Down-Liste, in der man von den Vergleichsoperatoren „<“, „<=“, „=“, „=>“, „>“ einen auswählen kann. Weiterhin noch eine Drop-Down-Liste, in der man die Kantenrichtung auswählen kann („in“, „out“, „any“). Schließlich gibt es noch ein Eingabefeld für die Anzahl.

Name

Es gibt ein Eingabefeld, in das man den Objektnamen angeben kann. Außerdem gibt es eine Checkbox „Search Entire Graph“. Ist diese ausgewählt, wird der gesamte Graph durchsucht, ansonsten nur das später zu wählende Graph-Window.

Node Type

Es gibt eine Liste, in der man beliebig viele Knotentypen auswählen kann. Zu dieser gibt es folgende Buttons:

- **All**-Button, mit dem alle Kantentypen ausgewählt werden können.
- **Clear**-Button, mit dem alle Kantentypen deselektiert werden Können.
- **Complement**-Button, mit dem die Auswahl invertiert werden kann.

Arc Type

Es gibt eine Liste, in der man beliebig viele Kantentypen auswählen kann. Zu dieser gibt es folgende Buttons:

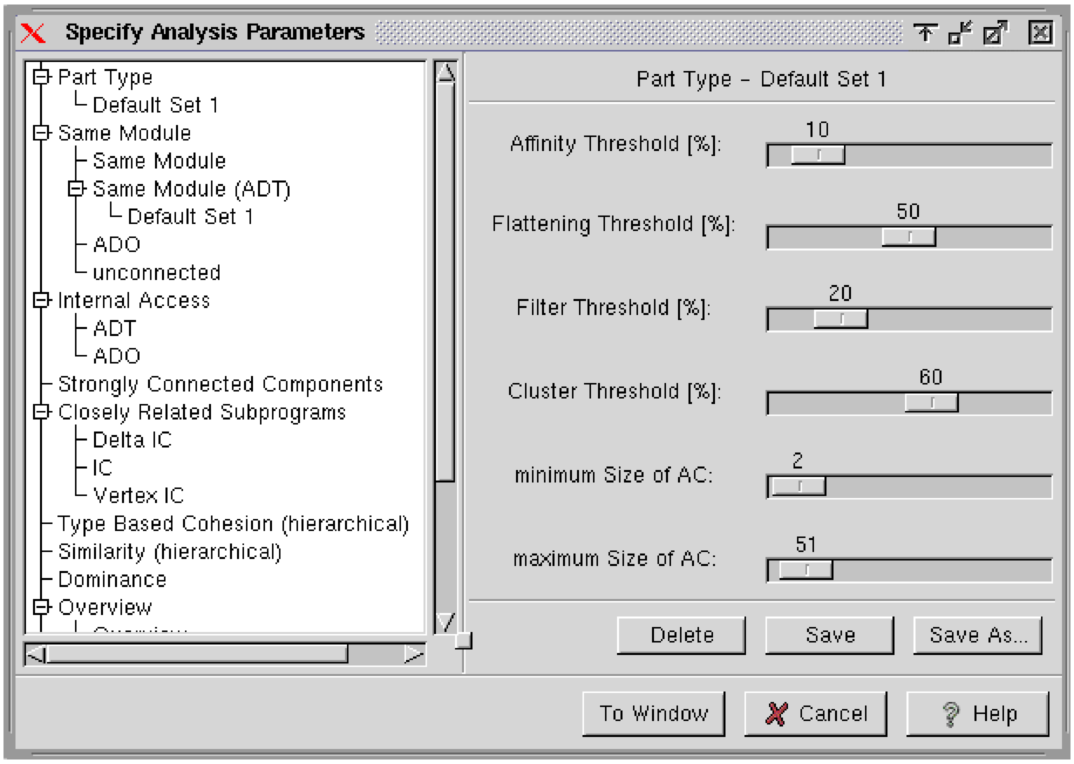
- **All**-Button, mit dem alle Kantentypen ausgewählt werden können.
- **Clear**-Button, mit dem alle Kantentypen deselektiert werden Können.
- **Complement**-Button, mit dem die Auswahl invertiert werden kann.

- Der **Delete**-Button löscht den links ausgewählten Parametersatz.
- Der **Save**-Button speichert die aktuell rechts eingestellten Kriterien unter dem links selektierten Parametersatz. Die alten Kriterien werden dabei überschrieben.
- Der **Save As...**-Button speichert einen im rechten Feld eingestellten Parametersatz unter einem neuen Namen ab. Der bisherige Parametersatz bleibt allerdings unter dem bisherigen Namen erhalten. Der Dialog *Parameter Set Name* (Abschnitt namens

Parameter Set Name) wird geöffnet. Nachdem dort der Name eingegeben wurde, wird der Parametersatz gespeichert und anschließend im linken Feld angezeigt.

- Der **Select Window**-Button ändert den Mauszeiger in den Select-Pointer. Nachdem der Benutzer ein Graph-Window ausgewählt hat, werden alle RFG-Objekte im aktuellen Graph-Window, die den Kriterien entsprechen selektiert.
- Der **Filter Selection**-Button ändert den Mauszeiger in den Filter-Selection-Pointer. Nachdem der Benutzer ein Graph-Window ausgewählt hat, wird mit dem im rechten Feld angegebenen Kriterien gefiltert.
- Der **Filter Window**-Button ändert den Mauszeiger in den Filter-Window-Pointer. Nachdem der Benutzer ein Graph-Window ausgewählt hat, wird mit dem im rechten Feld angegebenen Kriterien gefiltert.
- Der **Unfilter**-Button ändert den Mauszeiger in den Unfilter-Pointer. Nachdem der Benutzer ein Graph-Window ausgewählt hat, werden alle Filter in diesem Fenster rückgängig gemacht.
- Der **Cancel**-Button bricht die Funktion ab. Die Dialogbox wird nicht mehr angezeigt.

Analysis-Master



In diesem Dialog können die Analyseparameter verändert und Analysen gestartet werden.

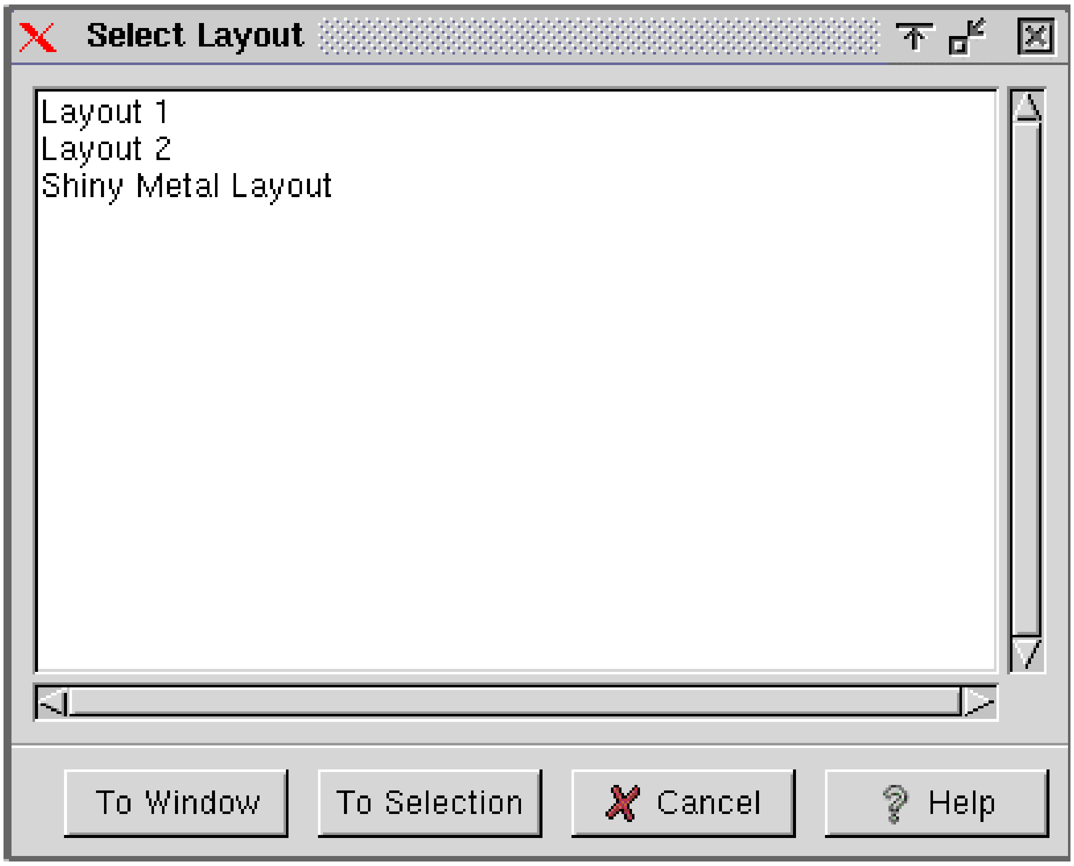
Abbildung 4-7. Dialog: Analysis-Master

- Im linken Feld kann eine Analyse oder ein Parametersatz zu einer Analyse ausgewählt werden. Der Eintrag wird dann gehighlighted. Beim ersten Öffnen des Masters am Anfang einer OpenCAGE-Sitzung ist automatisch der erste Eintrag selektiert. Während der OpenCAGE-Sitzung ist immer der Eintrag selektiert, den der Benutzer als letztes selektiert hat.
- Der **Delete**-Button löscht den links ausgewählten Parametersatz. Danach ist ein beliebiger Parametersatz ausgewählt.
- Der **Save**-Button speichert die aktuell rechts eingestellten Kriterien unter dem links selektierten Parametersatz. Die alten Kriterien werden dabei überschrieben.
- Der **Save As**-Button speichert einen im rechten Feld eingestellten Parametersatz unter einem neuen Namen ab. Der alte Parametersatz bleibt allerdings unter dem bisherigen Namen erhalten. Der Dialog *Parameter Set Name* (Abschnitt namens *Parameter Set Name*) wird geöffnet. Nachdem dort der Name eingegeben wurde, wird der Parametersatz gespeichert und anschließend im linken Feld angezeigt.
- Wenn im linken Feld eine Analyse ausgewählt wurde, ändert der **To Window**-Button den Mauszeiger in den Analyze-Window-Pointer. Nach der Wahl eines Graph-Windows startet diese Analyse mit dem im rechten Feld angezeigten Parametersatz.

  Ist links ein Parametersatz ausgewählt, ändert sich der Mauszeiger in den Analyze-Window-Pointer. Dann wird nach der Wahl eines Graph-Windows die zu diesem Satz gehörende Analyse mit dem im rechten Feld angezeigten Parametersatz (nicht notwendigerweise gleich dem abgespeicherten Satz!) gestartet. Dann wird der Dialog geschlossen und der Dialog *Progress* (Abschnitt namens *Progress*) angezeigt.

- Der **Cancel**-Button beendet den Dialog, ohne eine Analyse zu starten.

Layout-Master

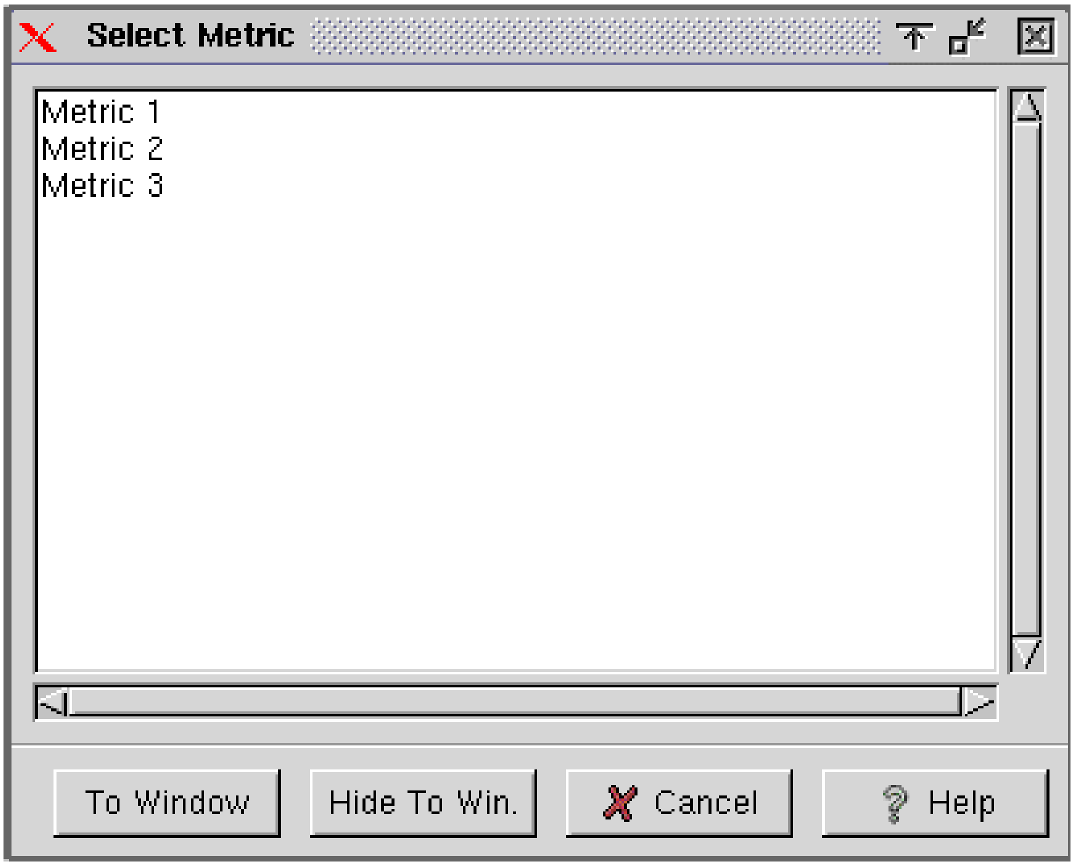


In diesem Master können Layouts ausgewählt und angewendet werden.

Abbildung 4-8. Dialog: Layout-Master

- Im oberen Teil des Dialogs befindet sich eine Liste der verfügbaren Layouts. Durch Anklicken eines Eintrags wird dieser selektiert. Beim ersten Öffnen des Masters am Anfang einer OpenCAGE-Sitzung ist automatisch der erste Eintrag selektiert. Während der OpenCAGE-Sitzung ist immer der Eintrag selektiert, den der Benutzer als letztes selektiert hat.
- Der **To Window**-Button ändert den Mauszeiger in den Layout-Window-Pointer. Nachdem der Benutzer ein Graph-Window ausgewählt hat, wird das selektierte Layout auf dieses Graph-Window angewendet.
- Der **To Selection**-Button ändert den Mauszeiger in den Layout-Selection-Pointer. Nachdem der Benutzer ein Graph-Window ausgewählt hat, wird das selektierte Layout auf die selektierten RFG-Objekte aus diesem Graph-Window angewendet.
- Der **Cancel**-Button beendet den Dialog, ohne ein Layout anzuwenden.

Metric-Master



In diesem Master kann der Benutzer eine Metrik auswählen und auf ein Graph-Window anwenden.

Abbildung 4-9. Metric-Master

- In der Liste kann der Benutzer auf einen Eintrag klicken. Der Eintrag wird dadurch selektiert.
- Der **To Window**-Button ändert den Mauszeiger in den Metrics-Window-Pointer. Nachdem der Benutzer ein Graph-Window ausgewählt hat, wird die selektierte Metrik auf alle Knoten dieses Graph-Windows angewendet.
- Der **Hide To Win.**-Button ändert den Mauszeiger in den Metrics-Window-Pointer. Nachdem der Benutzer ein Graph-Window ausgewählt hat, wird die Metrik bei allen Knoten dieses Graph-Windows wieder rückgängig gemacht.
- Der **Cancel**-Button beendet den Dialog erfolglos. Die Metrikvisualisierung bleibt gleich.

Dialoge

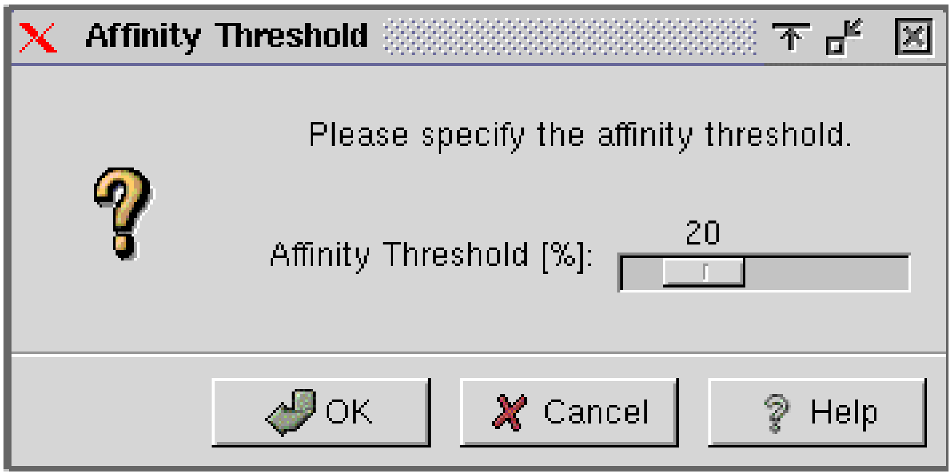
About

Dieser Dialog zeigt Informationen zu OpenCAGE an.

Abbildung 4-10. Dialog: About

- Der **OK**-Button schließt den Informationsdialog wieder.

Affinity Threshold

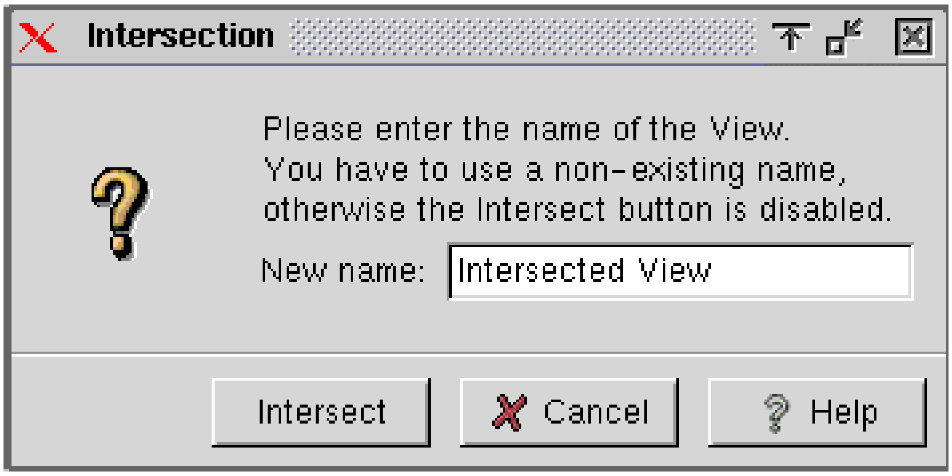


Dieser Dialog wird benutzt um den Wert des Affinity Threshold einzustellen.

Abbildung 4-11. Dialog: Affinity Threshold

- Mit dem Slider wird der Wert des Affinity Threshold im Wertebereich von 0 bis 100% in Schritten von einem Prozent eingestellt.
- Der **OK**-Button beendet den Dialog erfolgreich, d.h. die gemachten Änderungen des Affinity Threshold werden von OpenCAGE für spätere Operationen abgespeichert.
- Der **Cancel**-Button bricht den Dialog ab, d.h. es werden keine Änderungen am Affinity Threshold gespeichert.

Intersection

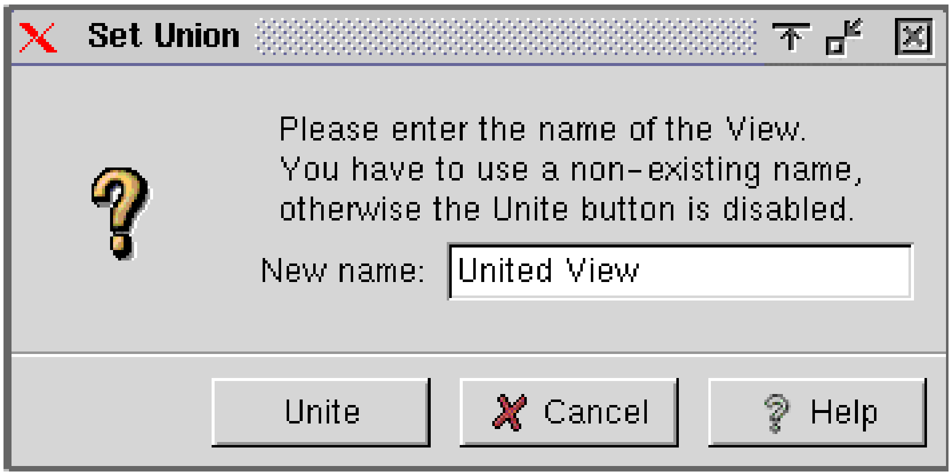


In diesem Dialog kann der Benutzer den Namen der neu zu erzeugenden Schnitt-View eingeben.

Abbildung 4-12. Dialog: Intersection

- Direkt nach Aufruf des Dialoges ist das Editierfeld leer. Das Editierfeld akzeptiert bis zu 256 der in Abschnitt namens *Texteingabefelder* in Kapitel 3 erlaubten Zeichen.
- Der **Intersect**-Button beendet den Dialog erfolgreich, d.h. der Schnitt wird durchgeführt. OpenCAGE testet nach jedem eingegebenen Zeichen, ob der Name als View-Name bereits vergeben ist. Existiert bereits eine View mit gleichem Namen, ist der **Intersect**-Button ausgegraut. Auch wenn das Editierfeld leer ist, ist der Button ausgegraut. Ansonsten kann der Benutzer den **Intersect**-Button drücken.
- Der **Cancel**-Button bricht den Dialog ab, d.h. der Schnitt wird nicht durchgeführt.

Set Union

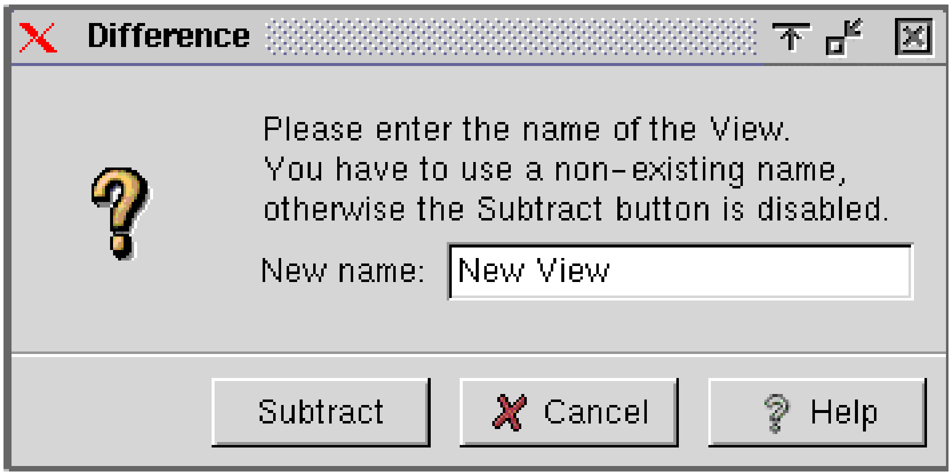


Im Set-Union-Dialog kann der Benutzer den Namen der neu zu erzeugenden Vereinigungs-View eingeben.

Abbildung 4-13. Dialog: Set Union

- Direkt nach Aufruf des Dialoges ist das Editierfeld leer. Das Editierfeld akzeptiert bis zu 256 der in Abschnitt namens *Texteingabefelder* in Kapitel 3 erlaubten Zeichen.
- Der **Unite**-Button beendet den Dialog erfolgreich, d.h. die Vereinigung wird durchgeführt. OpenCAGE testet nach jedem eingegebenen Zeichen, ob der Name als View-Name bereits vergeben ist. Existiert bereits eine View mit gleichem Namen, ist der **Unite**-Button ausgegraut. Auch wenn das Editierfeld leer ist, ist der Button ausgegraut. Ansonsten kann der Benutzer den **Unite**-Button drücken.
- Der **Cancel**-Button bricht den Dialog ab, d.h. es wird keine Vereinigung durchgeführt.

Difference

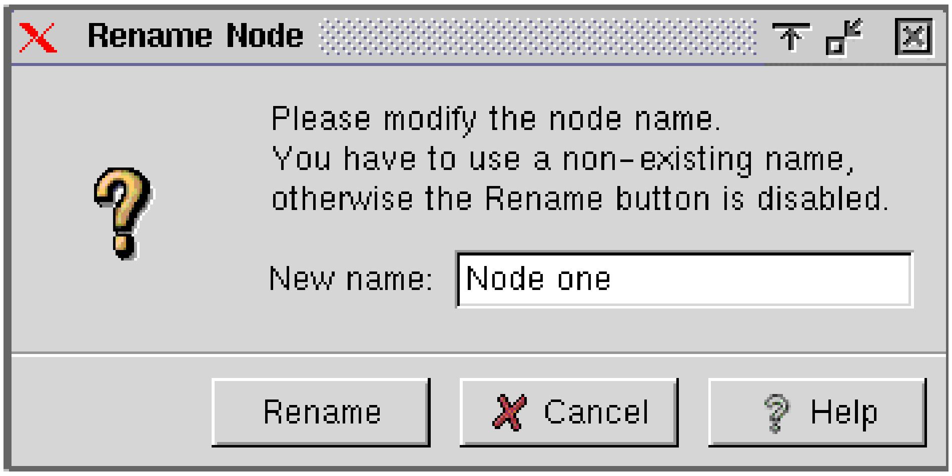


Im Difference-Dialog kann der Benutzer den Namen der neu zu erzeugenden Vereinigungs-View eingeben.

Abbildung 4-14. Dialog: Difference

- Direkt nach Aufruf des Dialoges ist das Editierfeld leer. Das Editierfeld akzeptiert bis zu 256 der in Abschnitt namens *Texteingabefelder* in Kapitel 3 erlaubten Zeichen.
- Der **Subtract**-Button beendet den Dialog erfolgreich, d.h. die Vereinigung wird durchgeführt. OpenCAGE testet nach jedem eingegebenen Zeichen, ob der Name als View-Name bereits vergeben ist. Existiert bereits eine View mit gleichem Namen, ist der **Subtract**-Button ausgegraut. Auch wenn das Editierfeld leer ist, ist der Button ausgegraut. Ansonsten kann der Benutzer den **Subtract**-Button drücken.
- Der **Cancel**-Button bricht den Dialog ab, d.h. es wird keine Differenzmenge gebildet.

Rename Node

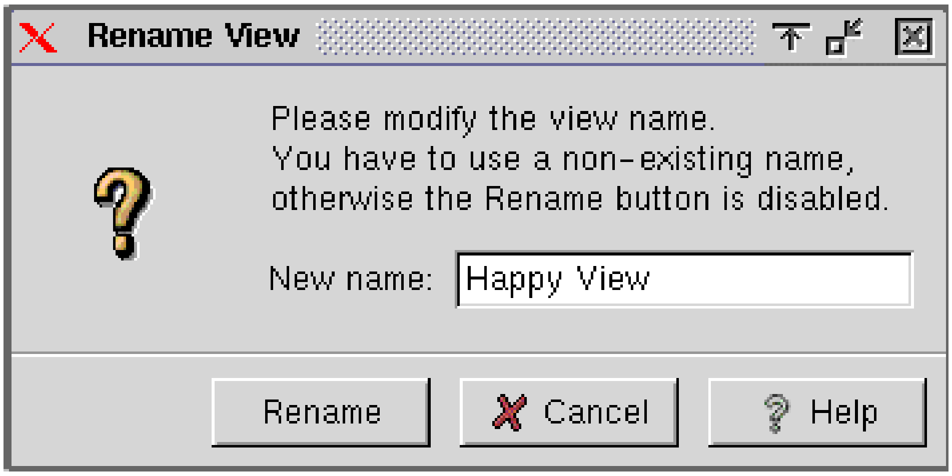


Dieser Dialog wird benutzt, um den Objektnamen eines Knoten umzubenennen. Die restlichen Bestandteile des Knotennamens bleiben unberührt.

Abbildung 4-15. Dialog: Rename Node

- Direkt nach Aufruf des Dialoges steht im Editierfeld der alte Knotenname. Das Editierfeld akzeptiert bis zu 256 der in Abschnitt namens *Texteingabefelder* in Kapitel 3 erlaubten Zeichen.
- Der **Rename**-Button beendet den Dialog erfolgreich, d.h. die Umbenennung des Knotens wird durchgeführt. Wenn das Editierfeld leer ist, ist der Button ausgegraut. Ansonsten kann der Benutzer den **Rename**-Button drücken.
- Der **Cancel**-Button bricht den Dialog ab, d.h. es werden keine Änderungen am Knotennamen gemacht.

Rename View

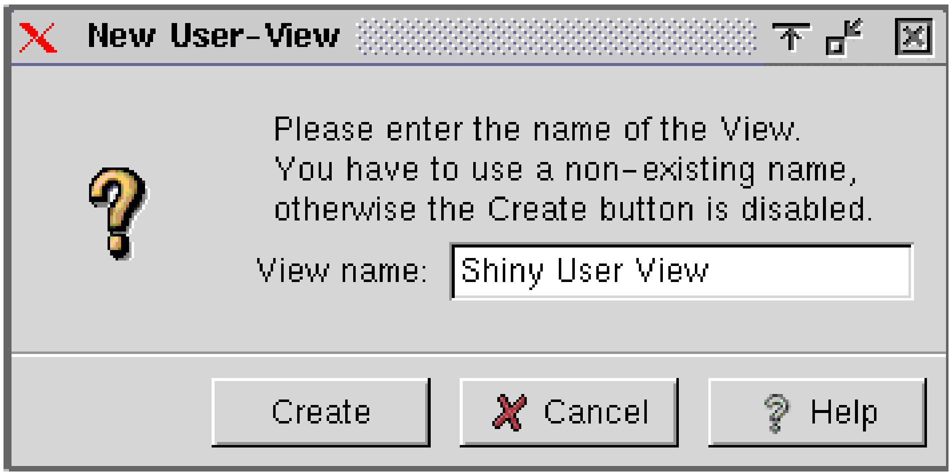


Dieser Dialog wird benutzt um eine View umzubenennen.

Abbildung 4-16. Dialog: Rename View

- Direkt nach Aufruf des Dialoges steht im im Editierfeld der alte Viewname. Das Editierfeld akzeptiert bis zu 256 der in Abschnitt namens *Texteingabefelder* in Kapitel 3 erlaubten Zeichen.
- Der **Rename**-Button beendet den Dialog erfolgreich, d.h. die Umbenennung der View wird durchgeführt. OpenCAGE testet nach jedem eingegebenen Zeichen, ob der Name als View-Name bereits vergeben ist. Existiert bereits eine View mit gleichem Namen, ist der **Rename**-Button ausgegraut. Auch wenn das Editierfeld leer ist, ist der Button ausgegraut. Ansonsten kann der Benutzer den **Rename**-Button drücken.
- Der **Cancel**-Button bricht den Dialog ab, d.h. es werden keine Änderungen am Viewnamen gemacht.

New User-View

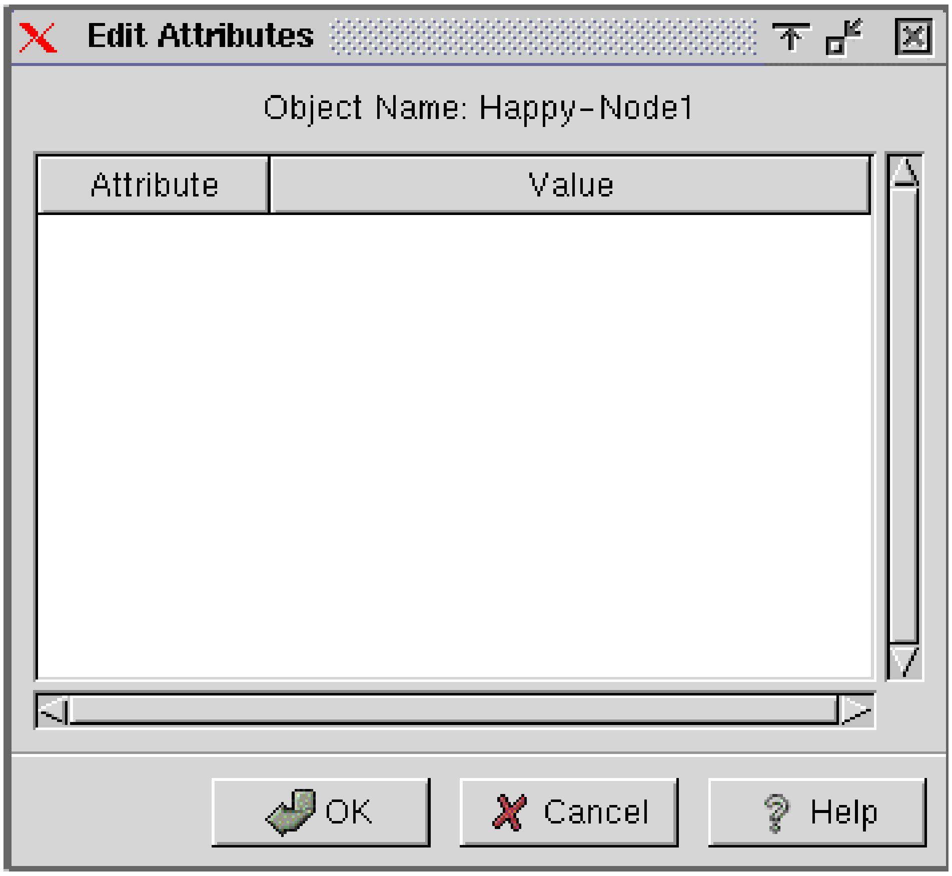


Dieser Dialog wird benutzt um den Namen der neuen User-View vom Benutzer abzufragen.

Abbildung 4-17. Dialog: New User-View

- Direkt nach Aufruf des Dialoges erscheint ein leeres Textfeld, in das der Benutzer den neuen Namen der User-View eingeben muss. Das Editierfeld akzeptiert bis zu 256 der in Abschnitt namens *Texteingabefelder* in Kapitel 3 erlaubten Zeichen.
- Der **Create**-Button beendet den Dialog erfolgreich, d.h. der Use-Case verwendet den eingegebenen Namen für die neue User-View. OpenCAGE testet nach jedem eingegebenen Zeichen, ob der Name als View-Name bereits vergeben ist. Existiert bereits eine View mit gleichem Namen, ist der **Create**-Button ausgegraut. Auch wenn das Editierfeld leer ist, ist der Button ausgegraut. Ansonsten kann der Benutzer den **Create**-Button drücken.
- Der **Cancel**-Button bricht den Dialog ab, d.h. es wird keine neue User-View erzeugt.

Edit Attributes

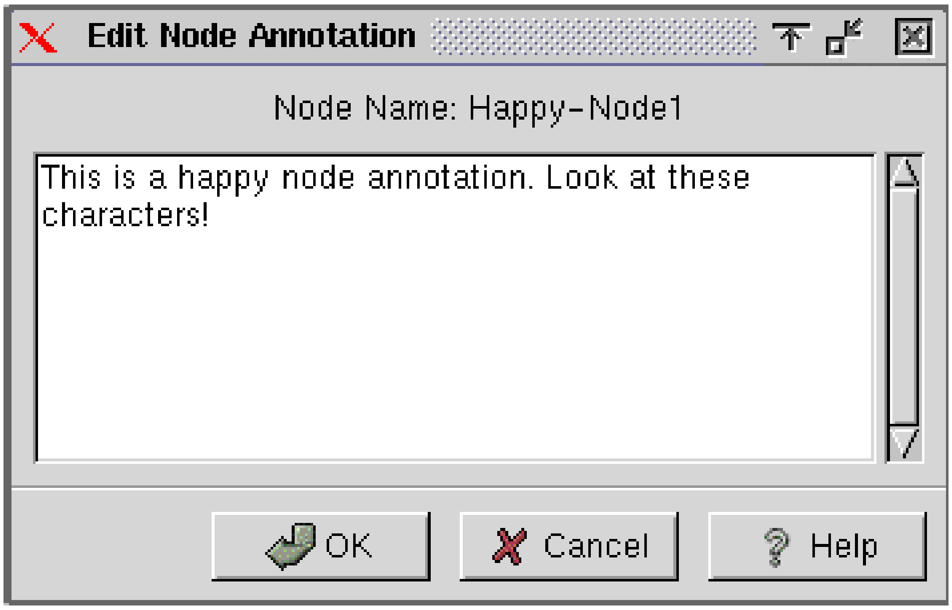


Dieser Dialog wird benutzt um Attribute eines RFG-Objektes zu ändern.

Abbildung 4-18. Dialog: Edit Attributes

- Direkt nach Aufruf des Dialoges erscheint in der weißen Fläche eine Tabelle, in der links alle Attribute des RFG-Objektes und rechts die entsprechenden Werte der Attribute stehen. Alle momentan im entsprechenden Graph-Window sichtbaren Attribute sind in der Tabelle eingefärbt. Der Benutzer kann in ein Value-Feld klicken und den Wert anschließend per Tastatur ändern.
- Der **OK**-Button beendet den Dialog erfolgreich, d.h. die gemachten Änderungen der Attributswerte des RFG-Objektes werden durchgeführt.
- Der **Cancel**-Button bricht den Dialog ab, d.h. es werden keine Änderungen der Attributswerte gespeichert.

Edit Node Annotation

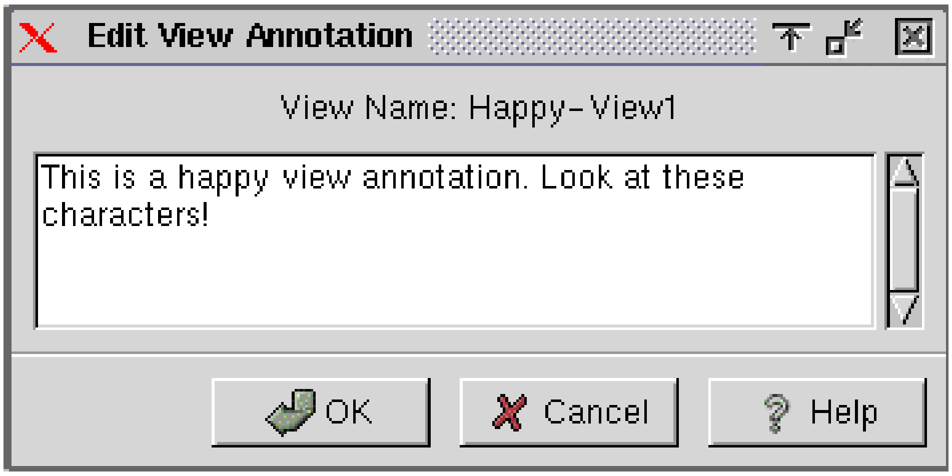


Dieser Dialog wird benutzt, um die Annotation eines Knotens zu ändern.

Abbildung 4-19. Dialog: Edit Node Annotation

- Direkt nach Aufruf des Dialoges erscheint in der weißen Fläche die bisherige Annotation des Knotens. Der Benutzer kann diese verändern.
- Der **OK**-Button beendet den Dialog erfolgreich, d.h. die gemachten Änderungen der Annotation werden durchgeführt.
- Der **Cancel**-Button bricht den Dialog ab, d.h. es werden keine Änderungen der Annotation gespeichert.

Edit View Annotation

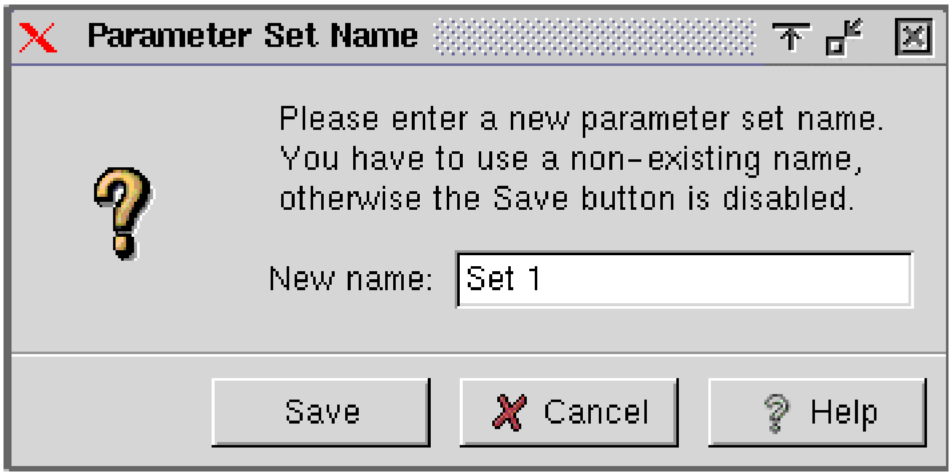


Dieser Dialog wird benutzt, um die Annotation einer View zu ändern.

Abbildung 4-20. Dialog: Edit View Annotation

- Direkt nach Aufruf des Dialoges erscheint in der weißen Fläche die bisherige Annotation der View. Der Benutzer kann diese verändern.
- Der **OK**-Button beendet den Dialog erfolgreich, d.h. die gemachten Änderungen der Annotation werden durchgeführt.
- Der **Cancel**-Button bricht den Dialog ab, d.h. es werden keine Änderungen der Annotation gespeichert.

Parameter Set Name

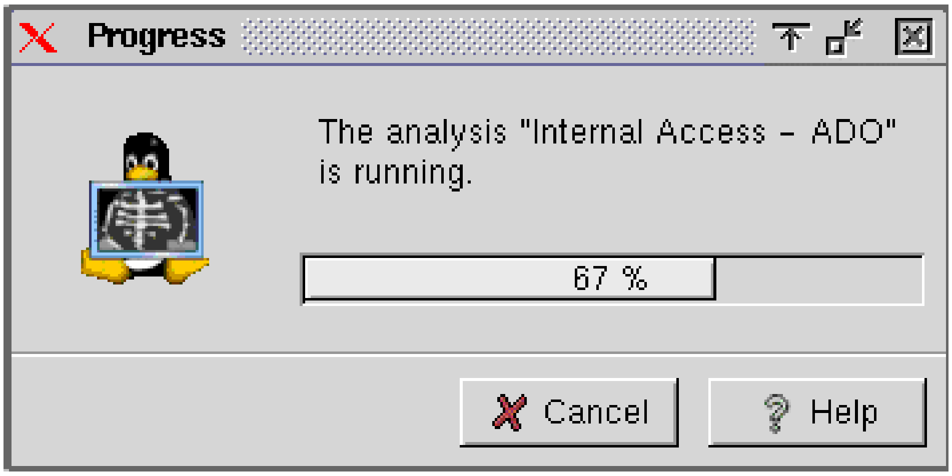


Dieser Dialog wird benutzt um einen neuen Namen für einen Parametersatz einzugeben.

Abbildung 4-21. Dialog: Parameter Set Name

- Direkt nach Aufruf des Dialoges ist das Editierfeld leer. Das Editierfeld akzeptiert bis zu 256 der in Abschnitt namens *Texteingabefelder* in Kapitel 3 erlaubten Zeichen.
- Der **Save**-Button beendet den Dialog erfolgreich, d.h. der Parametersatz wird unter diesem Namen abgespeichert. OpenCAGE testet nach jedem eingegebenen Zeichen, ob der Name als Parametersatz-Name bereits vergeben ist. Existiert bereits ein Parametersatz mit gleichem Namen, ist der **Save**-Button ausgegraut. Auch wenn das Editierfeld leer ist, ist der Button ausgegraut. Ansonsten kann der Benutzer den **Save**-Button drücken.
- Der **Cancel**-Button bricht den Dialog ab, d.h. der Parametersatz wird nicht gespeichert.

Progress

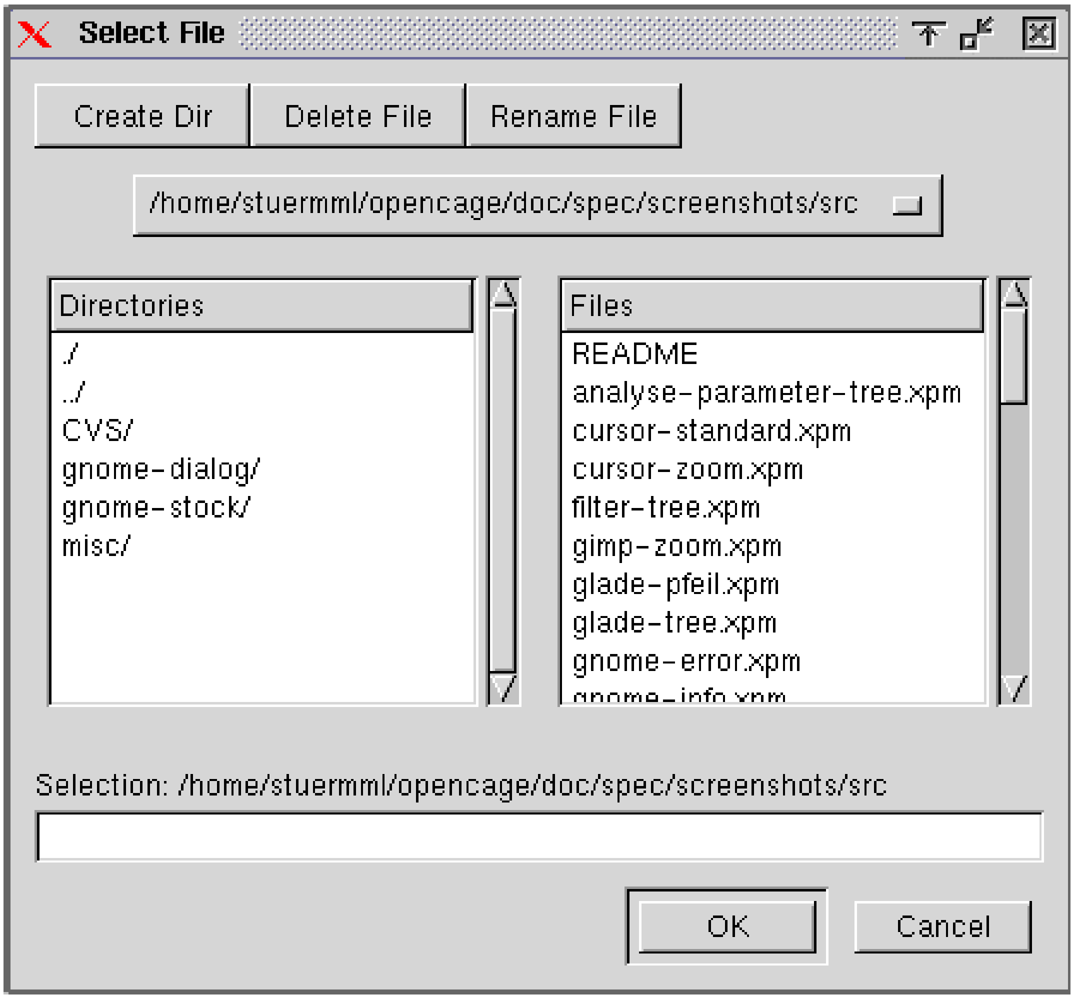


Dieser Dialog wird benutzt um den Fortschritt einer Operation anzuzeigen sowie die Operation abbrechen zu können.

Abbildung 4-22. Dialog: Progress

- Im Textfeld wird die gerade ablaufende Aktion beschrieben.
- Der Fortschrittsbalken zeigt den Fortschritt der Aktion in Prozent an. Wenn die genaue Dauer der Aktion nicht bekannt ist, wird stattdessen ein hin- und herwanderndes Feld im Balken angezeigt, der den Fortschritt der Aktion verdeutlicht.
- Der **Cancel**-Button bricht die Aktion ab. Je nach Aktion werden nicht alle Auswirkungen rückgängig gemacht (z.B. bei Layouts).

Select File



In diesem Dialog kann der Benutzer eine Datei auswählen.

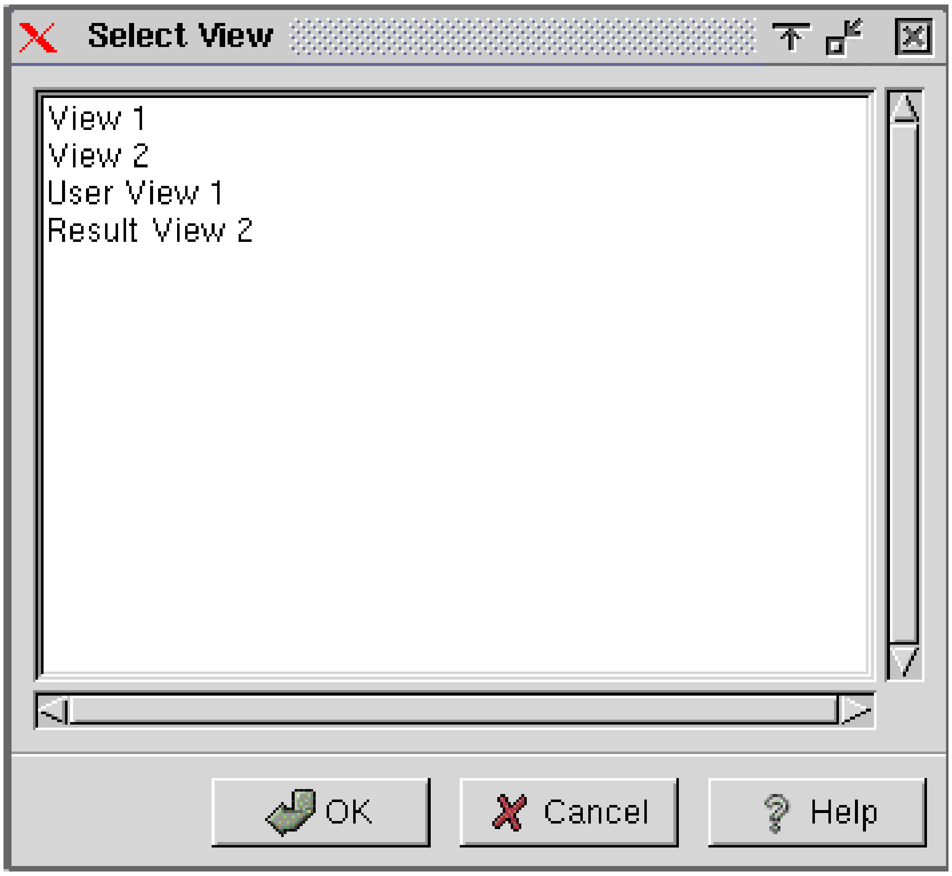
Die genaue Ausführung und Benutzung dieses Dialogs ist je nach Betriebssystem unterschiedlich. Hier wird der Dialog für gtk 1.2 unter Unix beschrieben.

Abbildung 4-23. Dialog: Select File

- Der **OK**-Button beendet den Dialog erfolgreich. Das heißt, die Datei wird ausgewählt.
- Der **Cancel**-Button beendet den Dialog erfolglos. Das heißt, die Auswahl-Operation wird abgebrochen.
- Der **Create Dir**-Button öffnet ein Dialogfenster, in dem der Benutzer den Namen des neuen Verzeichnisses angeben kann. Nachdem der Benutzer auf den **OK**-Button in diesem Dialogfenster klickt, erzeugt OpenCAGE ein neues leeres Verzeichnis mit diesem Namen, falls ein Verzeichnis oder eine Datei mit einen solchen Namen noch nicht existiert. Ansonsten zeigt es einen Fehler-Dialog an, dass eine solches Verzeichnis bereits existiert.

- Der **Delete File**-Button öffnet einen neuen Dialog, in dem der Benutzer gefragt wird, ob er wirklich die ausgewählte Datei löschen will. Klickt er auf den **OK**-Button, wird die ausgewählte Datei gelöscht. Ansonsten kehrt OpenCAGE - ohne die Datei zu löschen - zurück in den *Select File*-Dialog.
- Der **Rename File**-Button ändert einen Dateinamen. OpenCAGE öffnet einen Dialog, in denen der Benutzer den neuen Dateinamen eingeben kann. Klickt der Benutzer danach auf den **OK**-Button, überprüft OpenCAGE, ob der neue Dateiname bereits vergeben ist. Ist dies der Fall, erscheint ein Fehlerdialog. Existiert noch keine Datei und Verzeichnis und ist der Name gültig, dann wird die Datei umbenannt. Drückt der Benutzer auf den **Cancel**-Button, kehrt OpenCAGE ohne Namensänderung zurück in den *Select File*-Dialog.

Select one View



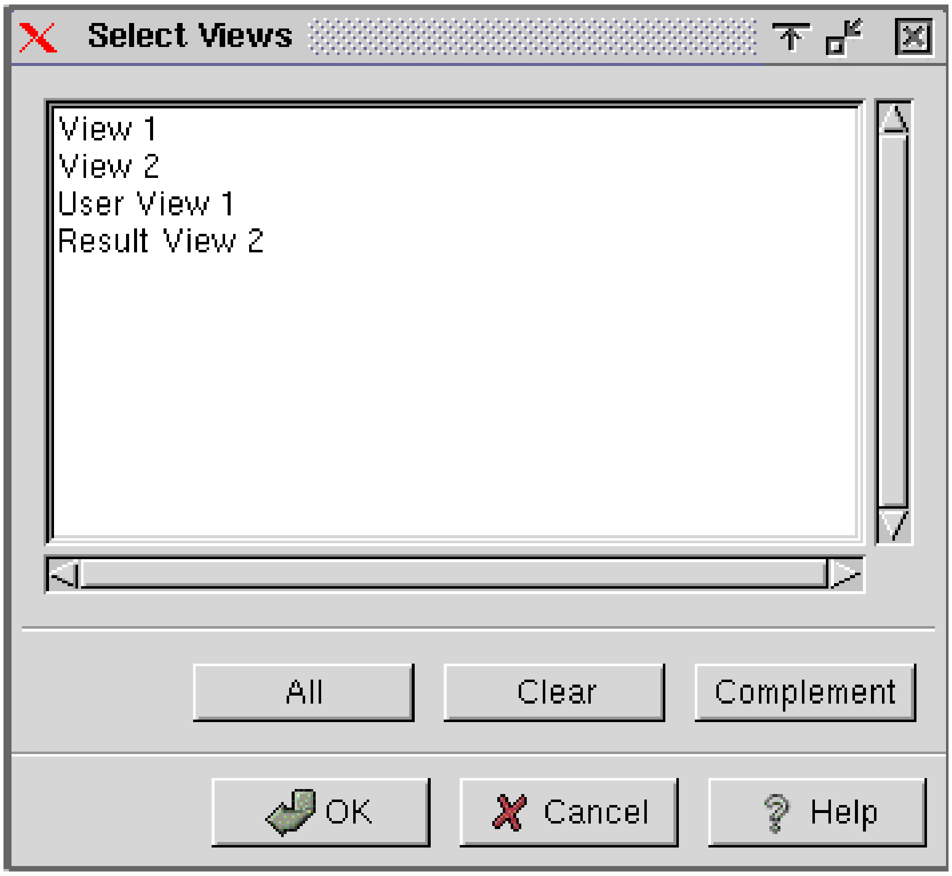
Dieser Dialog wird benutzt, um eine View auszuwählen.

Abbildung 4-24. Dialog: Select one View

- Direkt nach Aufruf des Dialoges erscheint in der weißen Fläche eine Liste mit Views, welche vom Use-Case abhängen. Der erste Eintrag ist ausgewählt. Der Benutzer kann durch einen Klick mit der linken Maustaste auf einen Eintrag diesen auswählen.

- Der **OK**-Button beendet den Dialog erfolgreich, d.h. der Use-Case verwendet die ausgewählte View.

Select Views



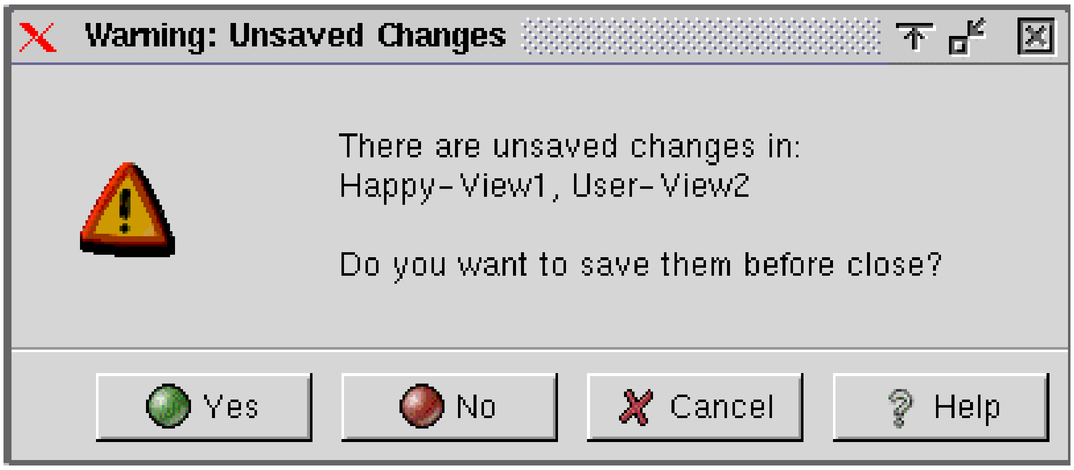
Dieser Dialog wird benutzt um eine Teilmenge aller Views auszuwählen.

Abbildung 4-25. Dialog: Select Views

- Direkt nach Aufruf des Dialoges erscheint in der weißen Fläche eine Liste mit Views, welche vom Use-Case abhängen. Der erste Eintrag ist ausgewählt. Der Benutzer kann durch einen Klick mit der linken Maustaste auf einen Eintrag diesen auswählen. Hält der Benutzer die **Strg**-Taste, während er auf einen Eintrag klickt, wird dieser zur gerade bestehenden Auswahl hinzugefügt. Hält der Benutzer die **Shift**-Taste, während er auf einen Eintrag klickt, werden alle Einträge, die sich zwischen der Auswahl und dem angeklickten Eintrag befinden, zur bestehenden Auswahl hinzugefügt.
- Der **OK**-Button beendet den Dialog erfolgreich, d.h. der Use-Case verwendet die ausgewählten Views.
- Der **Cancel**-Button bricht den Dialog ab, d.h. es werden keine Änderungen bei der Auswahl der Views gemacht.

- Der **All**-Button wählt alle Views in der Liste aus.
- Der **Clear**-Button markiert alle Views als nicht ausgewählt.
- Der **Complement**-Button wählt das Komplement der bisher ausgewählten Views aus.

Unsaved Changes

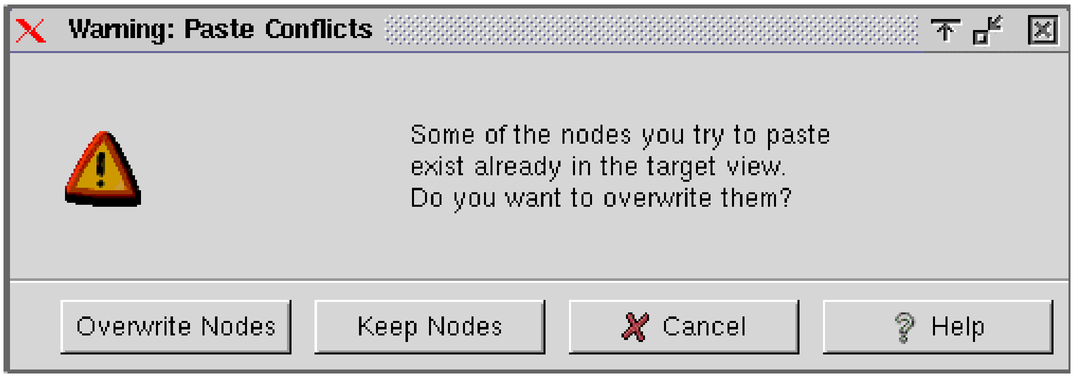


Dieser Dialog zeigt eine Warnung, dass ungesicherte Daten bestehen.

Abbildung 4-26. Dialog: Unsaved Changes

- Der **Yes**-Button beendet den Dialog und speichert die Änderungen ab. Danach werden alle Fenster außer der Workbench geschlossen.
- Der **No**-Button beendet den Dialog und schließt alle Fenster außer der Workbench, ohne die Änderungen zu speichern.
- Der **Cancel**-Button beendet den Dialog erfolglos. Das heißt, alle Fenster bleiben geöffnet.

Warning: Paste Conflict

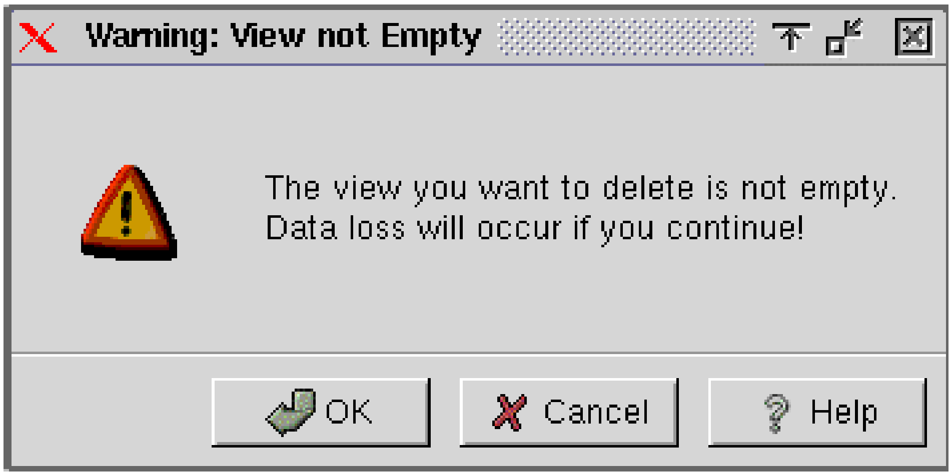


Dieser Dialog zeigt einen Konflikt beim Einfügen von Knoten an.

Abbildung 4-27. Dialog: Warning: Paste Conflict

- Der **Overwrite Nodes**-Button überschreibt die Knoten in der View, in die sie eingefügt werden mit den Knoten aus der Clipboard-View.
- Der **Keep Nodes**-Button lässt die Knoten in der View, in die eingefügt wird, wie sie sind und fügt die restlichen RFG-Objekte aus der Clipboard-View in die Zielview ein.
- Der **Cancel**-Button bricht die Aktion ab. Die Knoten, die den Konflikt verursachen werden durch Conflict-Highlighting graphisch hervorgehoben.

Warning: View not Empty

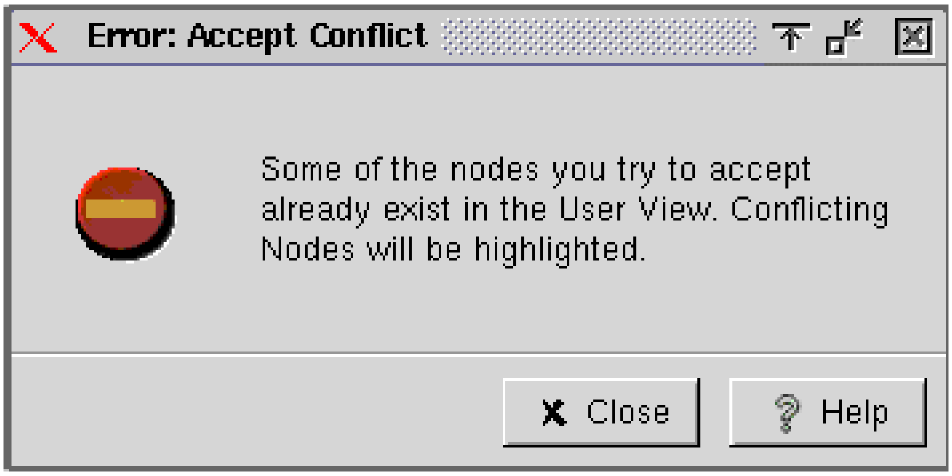


Dieser Dialog warnt den Benutzer, dass die zu löschende View nicht leer ist.

Abbildung 4-28. Dialog: Warning: View not Empty

- Der **OK**-Button beendet den Dialog erfolgreich. Das heißt die View wird gelöscht, obwohl sie noch RFG-Objekte enthält.
- Der **Cancel**-Button beendet den Dialog erfolglos. Das heißt, die View wird nicht gelöscht.

Error: Accept Conflict

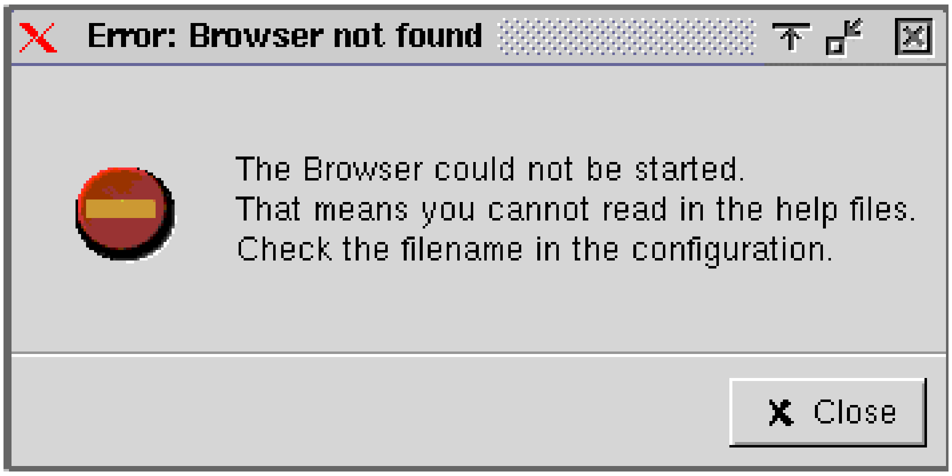


Dieser Dialog zeigt einen Konflikt beim Akzeptieren von Knoten an.

Abbildung 4-29. Dialog: Error: Accept Conflict

- Der **Close**-Button beendet den Dialog. Der Benutzer kann dann die Konflikte von Hand auflösen.

Error: Browser not found

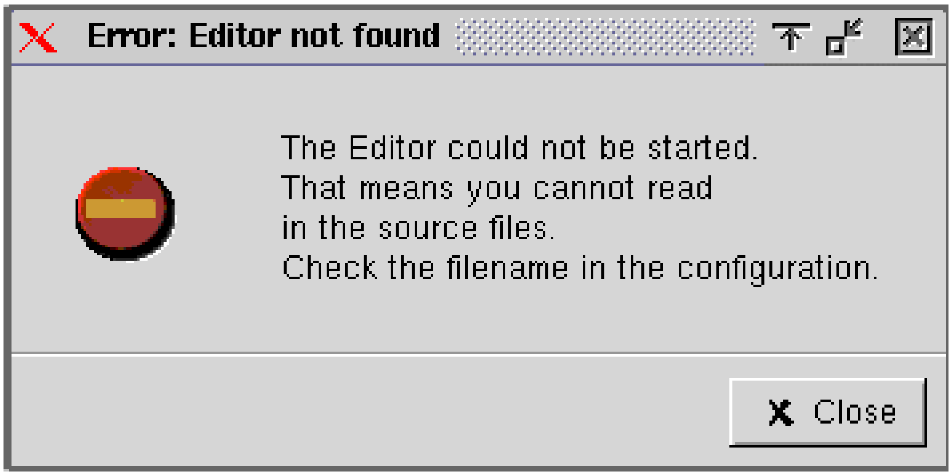


Dieser Dialog zeigt an, dass der Hilfebrowser nicht gefunden wurde oder nicht ausführbar ist.

Abbildung 4-30. Dialog: Error: Browser not found

- Im Text steht der Name des Browsers mit entsprechendem Pfad (siehe *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* (Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* in Kapitel 2)).
- Der **Close**-Button beendet den Dialog. Der Browser wird nicht gestartet.

Error: Editor not found



Dieser Dialog zeigt an, dass der Editor nicht gefunden wurde oder nicht ausführbar ist.

Abbildung 4-31. Dialog: Error: Editor not found

- Im Text steht der Name des Editors mit entsprechendem Pfad (siehe *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* (Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* in Kapitel 2)).
- Der **Close**-Button beendet den Dialog. Der Editor wird nicht gestartet.

Kapitel 5. Use Cases

Im Folgenden werden alle Use-Cases von OpenCAGE beschrieben. Sie werden dabei wie folgt aufgeteilt:

- *Select and Mark* (Abschnitt namens *Select and Mark*) ⇒ (Un)Select / (Un)mark - Aktionen in einem Graph-Window
- *Non RFG manipulation* (Abschnitt namens *Non RFG manipulation*) ⇒ Aktionen, die den RFG nicht manipulieren
- *RFG manipulation* (Abschnitt namens *RFG manipulation*) ⇒ Aktionen, die den RFG manipulieren, die von einem Graph-Window aus ausgeführt werden können
- *View-Master* (Abschnitt namens *View-Master*) ⇒ Aktionen, die aus dem *View-Master* (Abschnitt namens *View-Master* in Kapitel 4) aus ausgeführt werden können
- *Window-Master* (Abschnitt namens *Window-Master*) ⇒ Aktionen, die aus dem *Window-Master* (Abschnitt namens *Window-Master* in Kapitel 4) aus ausgeführt werden können
- *Workbench* (Abschnitt namens *Workbench*) ⇒ Aktionen, die nur aus dem *Workbench* (Abschnitt namens *Workbench* in Kapitel 4) aus ausgeführt werden können

Dabei gilt für alle Uses-Cases, ausgenommen *Open RFG* (Abschnitt namens *Open RFG*) und *Run Script* (Abschnitt namens *Run Script*), dass ein RFG bereits geöffnet ist.

Select and Mark

Die folgenden Use-Cases beschreiben Operationen, die für das Markieren und Selektieren in Graph-Windows von RFG-Objekten notwendig sind.

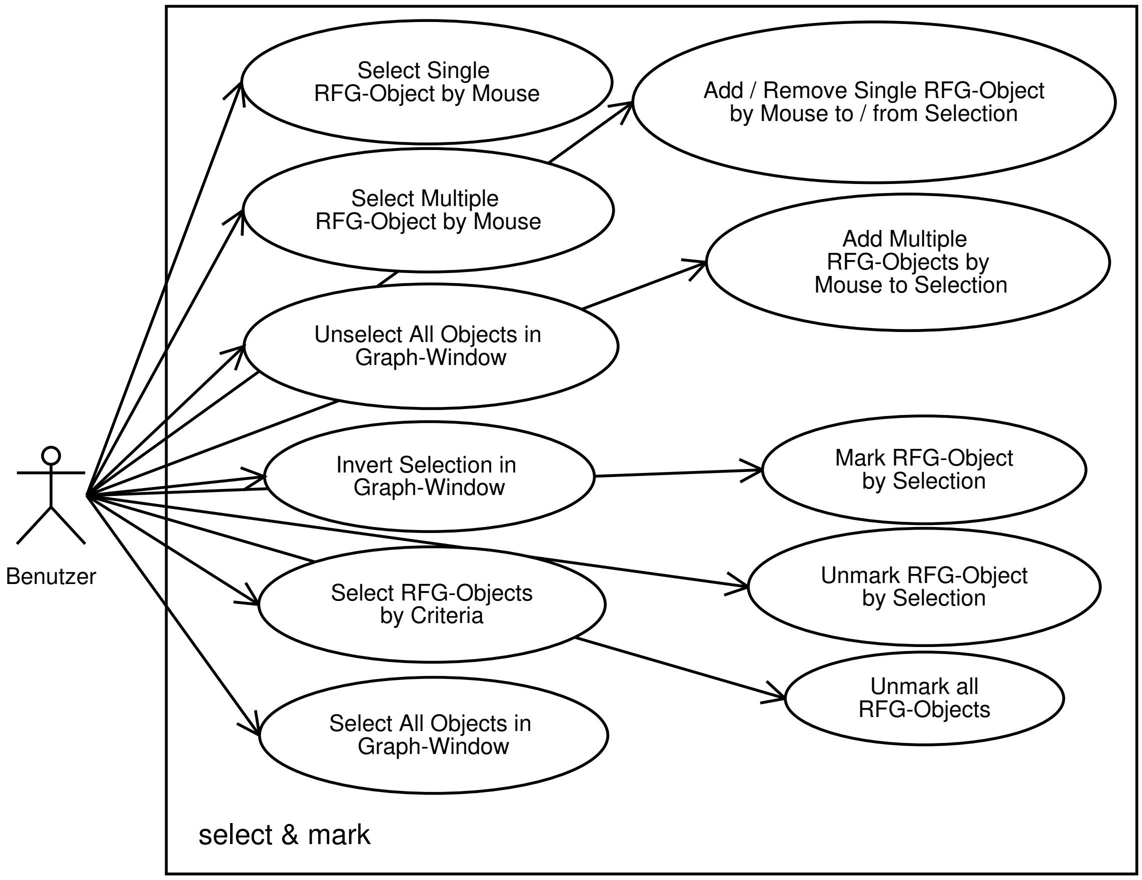


Abbildung 5-1. Use-Case-Diagramm: mark/select

Select Single RFG-Object by Mouse

Das Selektieren eines RFG-Objekts mit der Maus.

Vorbedingungen

- Mindestens ein RFG-Objekt muss in einem Graph-Window angezeigt werden.
- Der Selection-Mode muss aktiv sein. Das heißt, der Mauszeiger wird als Standard-Pointer angezeigt.

mögliche Einstiegspunkte

- Direkte Manipulation
- Skript

Ablauf

1. Der Benutzer klickt mit der linken Maustaste auf ein RFG-Objekt.

2. Alle RFG-Objekte, die bereits selektiert sind, werden deselektiert. OpenCAGE highlightet das RFG-Objekt, auf das der Benutzer geklickt hat, um anzuzeigen, dass es selektiert ist.

Select Multiple RFG-Objects by Mouse

Das Selektieren mehrerer RFG-Objekte mit der Maus.

Vorbedingungen

- Mindestens ein RFG-Objekt wird im aktuellen Graph-Window angezeigt.
- Der Selection-Mode muss aktiv sein. Das heißt, der Mauszeiger wird als Standard-Pointer angezeigt.

mögliche Einstiegspunkte

- Direkte Manipulation
- Skript

Ablauf

1. Der Benutzer drückt die linke Maustaste auf den Hintergrund der Graph-Area des aktuellen Graph-Windows und hält diese gedrückt.
2. OpenCAGE deselektiert alle bereits selektierten RFG-Objekte.
3. Der Benutzer bewegt den Pointer über das Graph-Window.
4. OpenCAGE zeichnet einen rechteckigen Rahmen, dessen einer Eckpunkt an der Stelle positioniert ist, wo der Benutzer die linke Maustaste gedrückt hat. Der andere Eckpunkt befindet sich an der aktuellen Position des Pointers. Der rechteckige Rahmen folgt dem Pointer.
5. Der Benutzer lässt die linke Maustaste los.
6. Der Rahmen verschwindet. Alle RFG-Objekte, die sich innerhalb des Rahmens befunden haben, werden gehighlightet, um anzuzeigen, dass sie selektiert sind.

Select All Objects in Graph Windows

Selektieren aller RFG-Objekte eines Graph-Windows

Vorbedingungen

- Mindestens ein RFG-Objekt muss in einem Graph-Window angezeigt werden.
- Der Selection-Mode muss aktiv sein. Das heißt, der Mauszeiger wird als Standard-Pointer angezeigt.

mögliche Einstiegspunkte

- Graph-Window-Button
- Skript

Ablauf

1. OpenCAGE deselektiert alle RFG-Objekte, die bereits selektiert sind.
2. OpenCAGE selektiert alle RFG-Objekte in einem Graph-Window. Die selektierten RFG-Objekte werden gehighlightet.

Invert Selection

Umkehren der Selektion in einem Graph-Window

Vorbedingungen

- Mindestens ein RFG-Objekt muss in einem Graph-Window angezeigt werden.
- Der Selection-Mode muss aktiv sein. Das heißt, der Mauszeiger wird als Standard-Pointer angezeigt.

mögliche Einstiegspunkte

- Graph-Window-Buttons
- Skript

Ablauf

1. OpenCAGE selektiert alle RFG-Objekte, die noch nicht selektiert sind und deselektiert alle, die vor Aufrufen des Use-Cases selektiert waren.

Select RFG-Objects by Criteria

Selektiert die RFG-Objekte, die bestimmten Kriterien entsprechen.

Vorbedingungen

- Es existiert ein Graph-Window.

mögliche Einstiegspunkte

- Menüleiste
- Masterbar
- Skript

Ablauf

1. OpenCAGE öffnet den *Filter/Select-Master* (Abschnitt namens *Filter/Select-Master* in Kapitel 4).
2. Der Benutzer wählt die gewünschten Parameter aus.
3. Der Benutzer drückt den **Select**-Button .
4. Der Benutzer bewegt den Pointer in eine Graph-Area und klickt die linke Maustaste.
5. OpenCAGE wendet die Kriterien auf die RFG-Objekte des aktuellen Graph-Windows an. Das heißt alle RFG-Objekte, die den Kriterien entsprechen, werden selektiert. Entspricht eine Kante den Kriterien, nicht aber der Quell- und auch nicht der Zielknoten, so wird auch diese Kante selektiert.

Alternative Abläufe

erster alternativer Ablauf

Bedingungen:

- Der Benutzer bewegt den Pointer über ein anderes OpenCAGE- Window.

1. Der Filter-Window-Pointer wird durch den No-Pointer ersetzt, um anzuzeigen, dass dieses Fenster kein Layout annehmen kann.
2. Klickt der Benutzer nun die linke Maustaste, dann wird der No-Pointer wieder durch den Standard-Pointer ersetzt und der Use-Case abgebrochen.

zweiter alternativer Ablauf

Bedingungen:

- Der Benutzer bewegt den Pointer über ein Fenster, das nicht zu OpenCAGE gehört

1. OpenCAGE ändert den Mauszeiger in den Standard-Pointer.

Unselect All RFG-Objects by Mouse

Deselektieren aller RFG-Objekte mit der Maus

Vorbedingungen

- Mindestens ein RFG-Objekt in einem Graph-Window muss selektiert sein.
- Der Selection-Mode muss aktiv sein. Das heißt der Standard-Pointer wird angezeigt.

mögliche Einstiegspunkte

- Direkte Manipulation

Ablauf

1. Der Benutzer klickt mit der linken Maustaste auf den Hintergrund der Graph-Area.
2. OpenCAGE macht das Highlighting aller RFG-Objekte im betreffenden Graph-Window rückgängig, um anzuzeigen, dass sie nicht mehr selektiert sind.

Add / Remove Single RFG-Object by Mouse to / from Selection

Hinzufügen eines einzelnen RFG-Objektes zu einer Menge selektierter RFG-Objekte mit der Maus bzw. deselektieren eines einzelnen RFG-Objektes von einer Menge selektierter RFG-Objekte mit der Maus.

Vorbedingungen

- Mindestens ein RFG-Objekt wird in einem Graph-Window angezeigt.
- Der Selection-Mode muss aktiv sein. Das heißt, der Mauszeiger wird als Standard-Pointer angezeigt.

mögliche Einstiegspunkte

- Direkte Manipulation
- Skript

Ablauf

1. Der Benutzer drückt die **Shift**-Taste und hält sie gedrückt.
2. Der Benutzer klickt mit der linken Maustaste auf ein RFG-Objekt, das sich im aktuellen Graph-Window befindet.
3. Wenn das angeklickte RFG-Objekt schon selektiert ist, dann wird es jetzt deselektiert und das Select-Highlighting entfernt.

   Wenn das angeklickte RFG-Objekt noch nicht selektiert ist, dann wird es jetzt selektiert und gehighlightet.

Add Multiple RFG-Objects by Mouse to Selection

Hinzufügen mehrerer RFG-Objekte zu einer Menge selektierter RFG-Objekte mit der Maus.

Vorbedingungen

- Mindestens ein RFG-Objekt wird in einem Graph-Window angezeigt.
- Mindestens ein RFG-Objekt ist selektiert.
- Der Selection-Mode muss aktiv sein. Das heißt, der Mauszeiger wird als Standard-Pointer angezeigt.

mögliche Einstiegspunkte

- Direkte Manipulation
- Skript

Ablauf

1. Der Benutzer drückt die **Shift**-Taste und hält diese gedrückt. Der Benutzer klickt mit der linken Maustaste auf den Hintergrund der Graph-Area des aktuellen Graph-Windows und hält diese ebenfalls gedrückt. Dann bewegt er den Pointer über die zu selektierenden RFG-Objekte im aktuellen Graph-Window.
2. OpenCAGE zeichnet einen rechteckigen Rahmen, dessen einer Eckpunkt an der Stelle positioniert ist, wo der Benutzer die linke Maustaste gedrückt hat. Der andere Eckpunkt befindet sich an der aktuellen Position des Pointers. Der rechteckige Rahmen folgt dem Pointer.
3. Der Benutzer lässt die linke Maustaste und die **Shift**-Taste los.
4. Der Rahmen verschwindet. Alle RFG-Objekte, die sich innerhalb des Rahmens befunden haben, werden gehighlightet, um anzuzeigen, dass sie selektiert sind. Bereits selektierte RFG-Objekte bleiben selektiert.

Alternative Abläufe

erster alternativer Ablauf

Bedingung: Der Benutzer lässt die **Shift**-Taste los, während der Rahmen gezeichnet wird. (Maustaste bleibt gedrückt).

1. Der Use-Case läuft regulär weiter.

Mark RFG-Object by Selection

Markieren von selektierten RFG-Objekten.

Vorbedingungen

- Mindestens ein RFG-Objekt muss in einem Graph-Window angezeigt werden.
- Mindestens ein RFG-Objekt muss selektiert sein.
- Der Selection-Mode muss aktiv sein. Das heißt, der Mauszeiger wird als Standard-Pointer angezeigt.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE hebt die selektierten RFG-Objekte durch Mark-Highlighting hervor, um anzuzeigen, dass diese markiert sind. Alle RFG-Objekte in anderen Graph-Windows, die im RFG-Sinne identisch zu den selektierten sind, werden ebenfalls markiert und durch Mark-Highlighting hervorgehoben.

Unmark All RFG-Objects

Rückgängigmachen der Markierungen aller RFG-Objekte.

Vorbedingungen

- Es ist mindestens ein RFG-Objekt markiert.
- Der Selection-Mode muss aktiv sein. Das heißt, der Mauszeiger wird als Standard-Pointer angezeigt.

mögliche Einstiegspunkte

- Kontextmenü
- Menüleiste
- Skript

Ablauf

1. OpenCAGE macht die Markierung und damit das Mark-Highlighting aller RFG-Objekte rückgängig.

Unmark RFG-Object by Selection

Löschen der Markierung von selektierten RFG-Objekten.

Vorbedingungen

- Mindestens ein RFG-Objekt muss in einem Graph-Window selektiert sein.
- Der Selection-Mode muss aktiv sein. Das heißt, der Mauszeiger wird als Standard-Pointer angezeigt.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE macht die Markierung und damit das Mark-Highlighting aller selektierter RFG-Objekte rückgängig.

Non RFG manipulation

Die folgenden Use-Cases beschreiben Operationen, die den RFG nicht manipulieren. Es sind im wesentlichen Use-Cases, die für die graphische Darstellung notwendig sind. (Un)Mark/(Un)Select-Aktionen werden unter *Select and Mark* (Abschnitt namens *Select and Mark*) beschrieben.

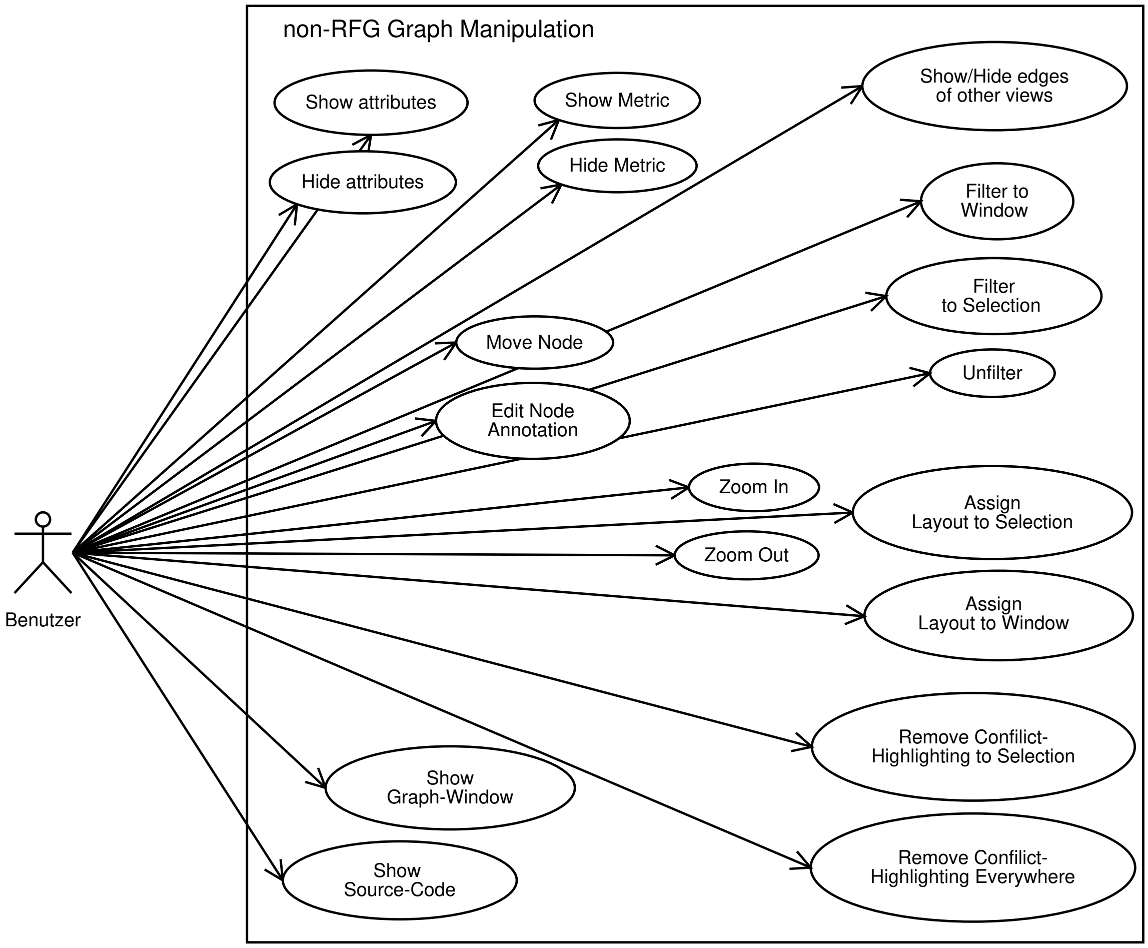


Abbildung 5-2. Use-Case-Diagramm: non RFG manipulation

Assign Layout to Selection

Richtet alle selektierten Knoten im aktuellen Graph-Window nach einem gewählten Layout aus.

Vorbedingungen

- Mindestens ein Knoten in einem Graph-Window ist selektiert.

mögliche Einstiegspunkte

- Menüleiste
- Masterbar
- Skript

Ablauf

1. OpenCAGE öffnet den Layout-Master *Layout-Master* (Abschnitt namens *Layout-Master* in Kapitel 4.
2. Der Benutzer wählt ein Layout aus.
3. Der Benutzer drückt den **To Selection** -Button.
4. OpenCAGE ändert den Mauszeiger in den Layout-Selection-Pointer (Abschnitt namens *Pointer* in Kapitel 4).
5. Der Benutzer bewegt den Pointer in eine Graph-Area und klickt die linke Maustaste.
6. OpenCAGE startet das entsprechende Layoutplugin.
   a. Die Layouts werden später in einem gesonderten Dokument beschrieben. Dieses wird erst in der entsprechenden Iterationsphase erstellt.
   b. Gegebenenfalls erfolgen weitere Interaktionen zwischen OpenCAGE und Benutzer. (Abfrage von einem Knoten, um den gruppiert werden soll)
7. Das Layoutplugin gruppiert die selektierten Knoten im gewählten Graph-Window neu. Die Kanten werden an die neue Gruppierung angepasst.

Alternative Abläufe

erster alternativer Ablauf

Bedingungen:

- Der Benutzer bewegt den Pointer über ein Fenster von OpenCAGE, das kein Graph-Window ist.

1. Der Layout-Selection-Pointer wird durch den No-Pointer ersetzt, um anzuzeigen, dass dieses Fenster kein Layout annehmen kann.
2. Klickt der Benutzer nun die linke Maustaste, dann wird der No-Pointer wieder durch den Standard-Pointer ersetzt und der Use-Case abgebrochen.

zweiter alternativer Ablauf

Bedingungen:

- Der Benutzer bewegt den Pointer über ein Fenster, das nicht zu OpenCAGE gehört

1. OpenCAGE ändert den Mauszeiger in den Standard-Pointer.

dritter alternativer Ablauf

Bedingungen:

- Der Benutzer klickt mit dem Layout-Selection-Pointer auf ein Graph-Window, in dem kein Knoten selektiert ist.

1. OpenCAGE ändert den Mauszeiger in den Standard-Pointer. Der Use-Case wird abgebrochen.

Assign Layout to Window

Richtet alle Knoten in einem zu wählenden Graph-Window nach einem gewählten Layout aus.

Vorbedingungen

- Es existiert ein Graph-Window.

mögliche Einstiegspunkte

- Menüleiste
- Masterbar
- Skript

Ablauf

1. OpenCAGE öffnet den Layout-Master *Layout-Master* (Abschnitt namens *Layout-Master* in Kapitel 4).
2. Der Benutzer wählt ein Layout aus.
3. Der Benutzer drückt den **To Window** Button.
4. OpenCAGE ändert den Mauszeiger in den Layout-Window-Pointer (Abschnitt namens *Pointer* in Kapitel 4).
5. Der Benutzer bewegt den Pointer in eine Graph-Area und klickt die linke Maustaste.
6. OpenCAGE startet das entsprechende Layoutplugin.
    a. Die Layouts werden später in einem gesonderten Dokument beschrieben. Dieses wird erst in der entsprechenden Iterationsphase erstellt.
    b. Gegebenenfalls erfolgen weitere Interaktionen zwischen OpenCAGE und Benutzer. (Abfrage von einem Knoten, um den gruppiert werden soll)
7. Das Layoutplugin gruppiert alle Knoten im gewählten Graph-Window neu. Die Kanten werden an die neue Gruppierung angepasst.

Alternative Abläufe

erster alternativer Ablauf

Bedingungen:

- Der Benutzer bewegt den Pointer über ein Fenster von OpenCAGE, das kein Graph-Window ist.

1. Der Layout-Window-Pointer wird durch den No-Pointer ersetzt, um anzuzeigen, dass dieses Fenster kein Layout annehmen kann.
2. Klickt der Benutzer nun die linke Maustaste, dann wird der No-Pointer wieder durch den Standard-Pointer ersetzt und der Use-Case abgebrochen.

zweiter alternativer Ablauf

Bedingungen:

- Der Benutzer bewegt den Pointer über ein Fenster, das nicht zu OpenCAGE gehört

1. OpenCAGE ändert den Mauszeiger in den Standard-Pointer.

Filter to Selection

Zeigt nur die RFG-Objekte einer Selektion an, die bestimmten Kriterien entsprechen.

Vorbedingungen

- Es existiert ein Graph-Window.

mögliche Einstiegspunkte

- Masterbar
- Menüleiste
- Skript

Ablauf

1. OpenCAGE öffnet den Filter-Master *Filter/Select-Master* (Abschnitt namens *Filter/Select-Master* in Kapitel 4.
2. Der Benutzer wählt ein Filter aus und trifft eventuell noch weitere Einstellungen.
3. Der Benutzer drückt den **To Selection** -Button.
4. OpenCAGE ändert den Mauszeiger in den Filter-Selection-Pointer.

5. Der Benutzer bewegt den Pointer in eine Graph-Area und klickt die linke Maustaste.
6. OpenCAGE wendet die Kriterien auf alle selektierten RFG-Objekte des aktuellen Graph-Windows an, indem nur noch solche RFG-Objekte angezeigt werden, die den Kriterien entsprechen. Kanten werden nur dann angezeigt, wenn Quell- und Zielknoten den Kriterien entsprechen. Entspricht nur einer der beiden Knoten den Kriterien, so wird sie als unvollständige Kante angezeigt. Alle anderen Kanten werden nicht angezeigt.

Alternative Abläufe

erster alternativer Ablauf

Bedingungen:

- Der Benutzer bewegt den Pointer über ein anderes OpenCAGE-Window.

1. Der Filter-Selection-Pointer wird durch den No-Pointer ersetzt, um anzuzeigen, dass dieses Fenster kein Layout annehmen kann.
2. Klickt der Benutzer nun die linke Maustaste, dann wird der No-Pointer wieder durch den Standard-Pointer ersetzt und der Use-Case abgebrochen.

zweiter alternativer Ablauf

Bedingungen:

- Der Benutzer bewegt den Pointer über ein Fenster, das nicht zu OpenCAGE gehört.

1. OpenCAGE ändert den Mauszeiger in den Standard-Pointer.

Filter to Window

Zeigt nur die RFG-Objekte an, die bestimmten Kriterien entsprechen.

Vorbedingungen

- Es existiert ein Graph-Window.

mögliche Einstiegspunkte

- Menüleiste
- Masterbar
- Skript

Ablauf

1. OpenCAGE öffnet den Filter-Master (Abschnitt namens *Filter/Select-Master* in Kapitel 4.
2. Der Benutzer wählt einen Filter aus und trifft eventuell noch weitere Einstellungen.
3. Der Benutzer drückt den Button **Filter Window**.
4. OpenCAGE ändert den Mauszeiger in den Filter-Window-Pointer.
5. Der Benutzer bewegt den Pointer in eine Graph-Area und klickt die linke Maustaste.
6. OpenCAGE wendet die Kriterien auf das aktuelle Graph-Window und zeigt nur noch die RFG-Objekte an, die den Kriterien entsprechen. Kanten werden nur dann angezeigt, wenn Quell- und Zielknoten den Kriterien entsprechen. Entspricht nur einer der Knoten den Kriterien, so wird die Kante als unvollständige Kante angezeigt. Alle anderen Kanten werden nicht angezeigt.

Alternative Abläufe

erster alternativer Ablauf

Bedingungen:

- Der Benutzer bewegt den Pointer über ein anderes OpenCAGE- Window.

1. Der Filter-Window-Pointer wird durch den No-Pointer ersetzt, um anzuzeigen, dass dieses Fenster kein Layout annehmen kann.
2. Klickt der Benutzer nun die linke Maustaste, dann wird der No-Pointer wieder durch den Standard-Pointer ersetzt und der Use-Case abgebrochen.

zweiter alternativer Ablauf

Bedingungen:

- Der Benutzer bewegt den Pointer über ein Fenster, das nicht zu OpenCAGE gehört

1. OpenCAGE ändert den Mauszeiger in den Standard-Pointer.

Unfilter

Hebt die Filterung von RFG-Objekten auf.

Vorbedingungen

- Es existiert ein Graph-Window.

mögliche Einstiegspunkte

- Menüleiste
- Masterbar
- Skript

Ablauf

1. OpenCAGE öffnet den Filter-Master (Abschnitt namens *Filter/Select-Master* in Kapitel 4.
2. Der Benutzer drückt den **Unfilter**-Button.
3. OpenCAGE ändert den Mauszeiger in den Unfilter-Pointer (siehe Abschnitt namens *Pointer* in Kapitel 4).
4. Der Benutzer bewegt den Pointer in eine Graph-Area und klickt die linke Maustaste.
5. OpenCAGE hebt alle Filter für das aktuelle Graph-Window auf und setzt das Filterstatus-Feld zurück.

Alternative Abläufe

erster alternativer Ablauf

Bedingungen:

- Der Benutzer bewegt den Pointer über ein anderes OpenCAGE-Window.

1. Es wird der No-Pointer angezeigt, um anzuzeigen, dass in diesem Fenster keine Filterung aufgehoben werden kann.
2. Klickt der Benutzer nun die linke Maustaste, dann wird der No-Pointer wieder durch den Standard-Pointer ersetzt und der Use-Case abgebrochen.

zweiter alternativer Ablauf

Bedingungen:

- Der Benutzer bewegt den Pointer über ein Fenster, das nicht zu OpenCAGE gehört.

1. OpenCAGE ändert den Mauszeiger in den Standard-Pointer.

Show attributes

Zeigt die Attribute an den Knoten im Graph-Window an.

Vorbedingungen

Es existiert ein aktuelles Graph-Window.

mögliche Einstiegspunkte

- Skript
- Kontextmenü

Ablauf

1. OpenCAGE blendet alle Attribute, die in der Konfigurationsdatei ausgewählt sind, zu allen Knoten des aktuellen Graph-Windows ein.

Hide attributes

Versteckt die Attribute an den Knoten im Graph-Window.

Vorbedingungen

- Es existiert ein aktuelles Graph-Window.

mögliche Einstiegspunkte

- Skript
- Kontextmenü

Ablauf

1. OpenCAGE blendet die Attribute aller Knoten im aktuellen Graph-Window aus.

Show Metric

Ändert die Knotengröße nach einer ausgewählten Metrik.

Vorbedingungen

- Es existiert ein Graph-Window.

mögliche Einstiegspunkte

- Menüleiste
- Masterbar
- Skript

Ablauf

1. OpenCAGE öffnet den Metric-Master *Metric-Master* (Abschnitt namens *Metric-Master* in Kapitel 4).
2. Der Benutzer drückt den **To Window** Button.
3. OpenCAGE ändert den Mauszeiger in den Metrics-Window-Pointer *Pointer* (Abschnitt namens *Pointer* in Kapitel 4).
4. Der Benutzer bewegt den Pointer in eine Graph-Area und klickt die linke Maustaste.
5. OpenCAGE ändert die Knotengröße aller Knoten im aktuellen Graph-Window nach der ausgewählten Metrik.

Alternative Abläufe

erster alternativer Ablauf

Bedingungen:

- Der Benutzer bewegt den Pointer über ein Fenster von OpenCAGE, das kein Graph-Window ist.

1. Der Metrics-Window-Pointer wird durch den No-Pointer ersetzt, um anzuzeigen, dass in diesem Fenster die Funktion nicht ausgeführt werden kann.
2. Klickt der Benutzer nun die linke Maustaste, dann wird der No-Pointer wieder durch den Standard-Pointer ersetzt und der Use-Case abgebrochen.

zweiter alternativer Ablauf

Bedingungen:

- Der Benutzer bewegt den Pointer über ein Fenster, das nicht zu OpenCAGE gehört.

1. OpenCAGE ändert den Mauszeiger in den Standard-Pointer.

Hide Metric

Ändert die Knotengröße zur Standardgröße.

Vorbedingungen

- Es existiert ein Graph-Window.

mögliche Einstiegspunkte

- Menüleiste
- Masterbar
- Skript

Ablauf

1. OpenCAGE öffnet den *Metric-Master* (Abschnitt namens *Metric-Master* in Kapitel 4).
2. Der Benutzer drückt den **Hide** Button.
3. OpenCAGE ändert den Mauszeiger in den Metrics-Window-Pointer (Abschnitt namens *Pointer* in Kapitel 4).
4. Der Benutzer bewegt den Pointer in eine Graph-Area und klickt die linke Maustaste.
5. OpenCAGE stellt die Knoten im aktuellen Graph-Window wieder in der Standardgröße dar.

Alternative Abläufe

erster alternativer Ablauf

Bedingungen:

- Der Benutzer bewegt den Pointer über ein Fenster von OpenCAGE, das kein Graph-Window ist.

1. Der Metrics-Window-Pointer wird durch den No-Pointer ersetzt, um anzuzeigen, dass in diesem Fenster die Funktion nicht ausgeführt werden kann.
2. Klickt der Benutzer nun die linke Maustaste, dann wird der No-Pointer wieder durch den Standard-Pointer ersetzt und der Use-Case abgebrochen.

zweiter alternativer Ablauf

Bedingungen:

- Der Benutzer bewegt den Pointer über ein Fenster, das nicht zu OpenCAGE gehört.

1. OpenCAGE ändert den Mauszeiger in den Standard-Pointer.

Remove Conflict-Highlighting to Selection

Macht das Conflict-Highlighting selektierter RFG-Objekte rückgängig.

Vorbedingungen

- Es existiert mindestens ein selektiertes RFG-Objekte mit Conflict-Highlighting.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE macht das Conflict-Highlighting aller selektierter RFG-Objekte rückgängig.

Remove Conflict-Highlighting Everywhere

Macht das Conflict-Highlighting aller RFG-Objekte rückgängig.

Vorbedingungen

- Es existiert mindestens ein RFG-Objekt mit Conflict-Highlighting.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE macht das Conflict-Highlighting aller RFG-Objekte in allen Graph-Windows rückgängig.

Move Nodes

Verschiebt Knoten innerhalb eines Graph-Windows.

Vorbedingungen

- Mindestens ein Knoten muss im aktuellen Graph-Window sichtbar sein.

mögliche Einstiegspunkte

- Direkte Manipulation
- Skript

Ablauf

1. Der Benutzer selektiert die zu verschiebende Knotenmenge.
2. Der Benutzer drückt mit der linken Maustaste auf die zu verschiebende Knotenmenge und hält die Taste gedrückt. Falls kein Knoten selektiert ist und der Benutzer klickt auf einen Knoten wird dieser automatisch selektiert.
3. Der Benutzer bewegt den Standard-Pointer an die Stelle innerhalb des aktuellen Graph-Windows, an die die Knotenmenge verschoben werden soll.
4. Beim Verschieben des Standard-Pointers erscheint unter dem Pointer die Kontur der Knotenmenge. Diese wird mit dem Pointer mitbewegt.
5. Der Benutzer lässt die linke Maustaste los.
6. Die Knotenmenge wird von ihrer ursprünglichen Position an die Position der Kontur verschoben. Die Kontur der Knotenmenge verschwindet.

Alternative Abläufe

erster alternativer Ablauf

Bedingungen:

- Der Benutzer verschiebt den Pointer über das Graph-Window hinaus.
- Der Pointer befindet sich nicht über der Graph-Area eines OpenCAGE-Fensters.

1. Der Standard-Pointer wird durch den No-Parking-Pointer ersetzt, um anzuzeigen, dass die Knotenmenge nicht an diese Stelle verschoben werden darf.
2. Lässt der Benutzer die linke Maustaste los, dann bleibt die Knotenmenge an ihrer ursprünglichen Position, die Kontur der Knotenmenge verschwindet und der No-Parking-Pointer wird durch den Standard-Pointer ersetzt.
3. Verschiebt der Benutzer den Pointer in die Graph-Area des Graph-Windows, in dem sich der Knoten befindet, dann wird der Use-Case ab Schritt 3 regulär abgearbeitet.
4. Verschiebt der Benutzer den Pointer in die Graph-Area eines Graph- Windows, in der sich die Knotenmenge nicht befindet, dann tritt keine Veränderung ein, d.h. es wird weiterhin der No-Parking-Pointer angezeigt und es ist weiterhin verboten, die Knotenmenge hierher zu verschieben. Allerdings ist in den Anforderungen, die den Rahmen dieses Projektes überschreiten, an dieser Stelle eine Drag&Drop-Funktionalität vorgesehen.

zweiter alternativer Ablauf

Bedingungen:

- Der Benutzer verschiebt den Pointer an den Rand des aktuellen Graph-Windows.

1. OpenCAGE scrollt die Graph-Area, sofern dies möglich ist.

Edit Node Annotation

Zeigt im Dialogfenster *Edit Node Annotation* (Abschnitt namens *Edit Node Annotation* in Kapitel 4) die Beschreibung des selektierten Knotens an.

Vorbedingungen

- Genau ein Knoten im aktuellen Graph-Window ist selektiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE öffnet den Dialog *Edit Node Annotation* (Abschnitt namens *Edit Node Annotation* in Kapitel 4).
2. Der Benutzer ändert den Beschreibungstest und klickt den **OK**-Button.
3. OpenCAGE speichert die neue Beschreibung.

Open Source-Code

Öffnet den Editor, der den Quellcode anzeigt, der dem Knoten zugeordnet ist.

Vorbedingungen

- Mindenstens ein Entitätsknoten muss im Graph-Window sichtbar sein.

mögliche Einstiegspunkte

- Direkte Manipulation
- Skript

Ablauf

1. Der Benutzer führt einen Doppelklick auf den zu öffnenden Entitätsknoten aus.
2. OpenCAGE selektiert den Knoten und öffnet den Editor, in dem der Quellcode, den der Knoten referenziert, angezeigt wird.

Alternative Abläufe

erster alternativer Ablauf

Bedingung: Der Editor wird nicht gefunden.

1. OpenCAGE zeigt den Fehlerdialog *Error: Editor not found* (Abschnitt namens *Error: Editor not found* in Kapitel 4) an.

Show Graph-Window

Öffnet ein neues Graph-Window, in dem alle Knoten und ihre induzierten Kanten angezeigt werden, die sich genau eine Hierarchie unter dem AC-Knoten befinden.

Vorbedingungen

- Mindestens ein AC-Knoten muss im aktuellen Graph-Window sichtbar sein.

mögliche Einstiegspunkte

- Direkte Manipulation
- Skript

Ablauf

1. Der Benutzer führt einen Doppelklick auf den zu öffnenden AC-Knoten aus.
2. OpenCAGE selektiert den Knoten.
3. OpenCAGE öffnet ein neues Graph-Window, in dem alle Knoten, die sich genau eine Hierarchie unter dem AC-Knoten befinden, und ihre induzierten Kanten angezeigt werden. Dabei wird der Style des aktuellen Graph-Windows übernommen.
4. OpenCAGE setzt dieses neue Graph-Window als aktuelles Graph-Window.

Show/Hide edges of other views

Zeigt oder versteckt die Kanten anderer Views im aktuellen Graph-Window.

Vorbedingungen

- Es existiert ein aktuelles Graph-Window.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE zeigt den Dialog *Select Views* (Abschnitt namens *Select Views* in Kapitel 4) an.
2. Der Benutzer wählt in einer Liste aller Views die Views aus, von denen die Kanten im aktuellen Graph-Window angezeigt werden sollen. Falls schon Kanten aus anderen Views angezeigt werden, sind diese Views in der Liste ausgewählt. Durch Klicken auf Listenelemente werden diese in der Markierung invertiert.
3. Der Benutzer klickt den **OK**-Button.
4. OpenCAGE zeigt bzw. verbirgt alle Kanten aus den ausgewählten Views im aktuellen Graph-Window. Ist nur einer der dieser Kante zugehörigen Knoten sichtbar, wird die Kante als unvollständige Kante angezeigt.

Zoom In

Zoomt in ein Graph-Window hinein

Vorbedingungen

- Es existiert ein Graph-Window.
- Der Zoom-Mode ist aktiv, das heisst, dass der Mauszeiger als Zoom-Pointer angezeigt wird.

mögliche Einstiegspunkte

- Werkzeugleiste
- Skript

Ablauf

1. Der Benutzer klickt mit der linken Maustaste in das Graph-Window, in das er hineinzoomen will.
2. OpenCAGE setzt dieses Fenster als aktuelles Graph-Window und zoomt aus diesem heraus. (Der Vergrösserungsfaktor wird in der Konfigurationsdatei festgelegt.)

Alternative Abläufe

erster alternativer Ablauf

Bedingung: Der Graph ist im Maßstab kleiner als 1:1 (bzgl. der Knoten-Bitmaps) sichtbar.

1. OpenCAGE behält den aktuellen Zoom bei.

Zoom Out

Zoomt aus einem Graph-Window heraus

Vorbedingungen

- Es existiert ein Graph-Window.
- Der Zoom-Mode ist aktiv, dass heisst dass der Mauszeiger als Zoom-Pointer angezeigt wird.

mögliche Einstiegspunkte

- Werkzeugleiste
- Skript

Ablauf

1. Der Benutzer klickt mit der linken Maustaste in das Graph-Window, aus dem er herauszoomen will.
2. OpenCAGE setzt dieses Fenster als aktuelles Graph-Window und zoomt aus diesem heraus. (Der Verkleinerungsfaktor wird in der Konfigurationsdatei festgelegt.)

Alternative Abläufe

erster alternativer Ablauf

Bedingung: Der komplette Graph ist in dem Graph-Window sichtbar.

1. OpenCAGE behält den aktuellen Zoom bei.

RFG manipulation

Die folgenden Use-Cases beschreiben Operationen, die den RFG manipulieren. Sie können in einem Graph-Window angewendet werden, es sei denn, dass die betreffende View durch den Benutzer explizit oder durch eine gestartete Analyse implizit auf read-only gesetzt wurde. Die Basis-Views sind permanent auf read-only gesetzt. Aktionen, die auf den RFG agieren, aber als Einstiegspunkt den *View-Master* enthalten werden unter *View-Master* (Abschnitt namens *View-Master* in Kapitel 4) beschrieben.

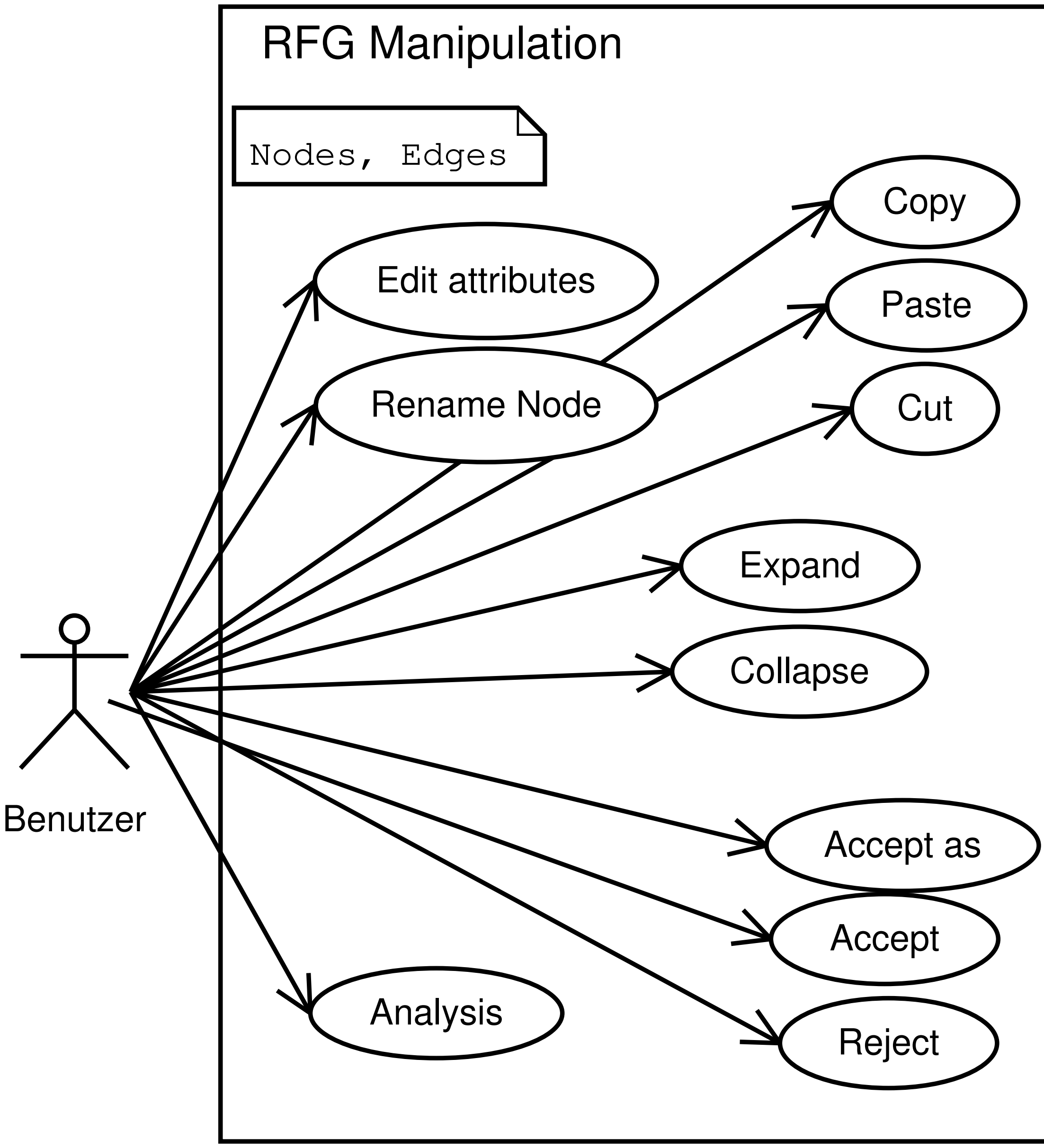


Abbildung 5-3. Use Case Diagramm: RFG manipulation

Accept

Akzeptiert einen oder mehrere selektierte Knoten (d.h. verschiebt diese in die aktuelle User-View). Falls alle Knoten eines Fensters akzeptiert werden sollen, müssen auch alle Knoten selektiert werden.

Vorbedingungen

- Es ist mindestens ein RFG-Objekt im Graph-Window selektiert.

- Es dürfen nur Kanten selektiert sein, deren Start- und Endknoten selektiert sind.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE übernimmt den oder die selektierten Knoten in die aktuelle User-View und entfernt diese samt ihren Kanten aus der aktuellen View. Falls im Moment ein Fenster, das diese User-View zeigt, geöffnet ist, dann holt OpenCAGE dieses in den Vordergrund und selektiert die neu hinzugekommenen RFG-Objekte. Bei AC-Knoten werden auch die darunterliegenden RFG-Objekte verschoben.

Accept as

Akzeptiert genau einen selektierten Knoten (d.h. verschiebt diesen in die aktuelle User-View). Davor kann der Benutzer den einzelnen Knoten umbenennen.

Vorbedingungen

- Es ist genau ein Knoten im aktuellen Graph-Window selektiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE öffnet den Dialog *Rename Node* (Abschnitt namens *Rename Node* in Kapitel 4)
2. Der Benutzer gibt den neuen Namen ein.
3. OpenCAGE übernimmt den selektierten Knoten in die aktuelle User-View und entfernt diesen samt seinen Kanten aus der aktuellen View. Falls im Moment ein Fenster, welches die User-View zeigt, geöffnet ist, holt OpenCAGE dieses in den Vordergrund und selektiert die neu hinzugekommenen RFG-Objekte.

Reject

Entfernt eine oder mehrere selektierte Kanten oder Knoten aus der aktuellen View und verschiebt diese in die Trashcan-View.

Vorbedingungen

- Es ist mindestens ein RFG-Objekt in einem Graph-Window selektiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE löscht alle RFG-Objekte aus der Trashcan-View.
2. Dann kopiert es in die Trashcan-View die selektierten Knoten, die Nachbarn der selektierten Knoten, die Kanten zwischen selektierten Knoten und ihren Nachbarknoten und die induzierten Kanten der selekierten Knoten.
3. OpenCAGE entfernt die selektierten Kanten und/oder die selektierten Knoten mit ihren induzierten Kanten aus der aktuellen View.

Copy

Kopiert die selektierten Knoten mit ihren induzierten Kanten aus der aktuellen View in die Clipboard-View.

Vorbedingungen

- Es ist mindestens ein Knoten selektiert.
- Es dürfen nur Kanten selektiert sein, deren Start- und Endknoten selektiert sind.
- Es sind nur RFG-Objekte aus demselben Graph-Window selektiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE löscht alle RFG-Objekte aus der Clipboard-View. Dann kopiert es die selektierten RFG-Objekte in die Clipboard-View.

Cut

Schneidet Knoten mit ihren induzierten Kanten aus der aktuellen View aus und fügt sie in die Clipboard-View ein.

Vorbedingungen

- Es ist mindestens ein Knoten selektiert.
- Es dürfen nur Kanten selektiert sein, deren Start- und Endknoten selektiert sind.
- Es sind nur RFG-Objekte aus einer View und aus einem Graph-Window selektiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE löscht alle RFG-Objekte aus der Clipboard-View.
2. Dann schneidet es die selektierten RFG-Objekte aus der aktuellen View aus und fügt sie in die Clipboard-View ein. Kanten zwischen nicht selektierten Knoten und selektierten Knoten, gehen dabei in der aktuellen View verloren.

Paste

Fügt die Knoten und Kanten aus der Clipboard-View in die aktuelle View ein.

Vorbedingungen

- Es ist ein Graph-Window geöffnet.
- In der Clipboard-View ist mindestens ein Knoten vorhanden.

mögliche Einstiegspunkte

- Kontextmenü

- Skript

Ablauf

1. OpenCAGE ändert das Aussehen des Mauszeigers in den Paste-Pointer.
2. Der Benutzer wählt im aktuellen Graph-Window die entsprechende Position aus, an der die Knoten und Kanten aus der Clipboard-View eingefügt werden sollen.
3. OpenCAGE fügt die Knoten und Kanten aus der Clipboard-View in das aktuelle Graph-Window an die Position des Mauszeigers ein.

Alternative Abläufe

erster alternativer Ablauf

Bedingungen:

- Im aktuellen Graph-Window existiert schon ein Teil der Knoten.
- Die Knoten im aktuellen Graph-Window haben andere Attribute als die Knoten, die eingefügt werden sollen.

1. Das OpenCAGE zeigt den Dialog *Warning: Paste Conflict* (Abschnitt namens *Warning: Paste Conflict* in Kapitel 4) an.
2. Mit Hilfe dieses Dialogs kann der Benutzer auswählen, ob die Knoten im aktuellen Graph-Window erhalten bleiben sollen und dafür die Knoten aus der Clipboard-View verloren gehen, ob die Knoten im aktuellen Graph-Window von den Knoten aus der Clipboard-View überschrieben werden oder ob der Vorgang abgebrochen wird.
3. Entscheidet sich der Benutzer dafür, den Vorgang abzubrechen, dann werden die Knoten, bei denen Konflikte aufgetreten sind, durch Conflict-Highlighting hervorgehoben.

Rename Node

Ändert den Namen eines Knotens.

Vorbedingungen

- Genau ein Knoten ist im aktuellen Graph-Window selektiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE öffnet den Dialog *Rename Node* (Abschnitt namens *Rename Node* in Kapitel 4).
2. OpenCAGE übernimmt den neuen Knotennamen im RFG und in der Darstellung.

Edit Attributes

Ändert die Attribute eines RFG-Objektes

Vorbedingungen

- Es ist genau ein RFG-Objekt im aktuellen Graph-Window selektiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE öffnet den Dialog *Edit Attributes* (Abschnitt namens *Edit Attributes* in Kapitel 4).
2. OpenCAGE hebt alle Attribute, die momentan sichtbar sind, graphisch hervor. OpenCAGE sortiert die Attributsliste alphabetisch.
3. Der Benutzer klickt in ein Value-Feld und ändert den Text dieses Feldes. Dieser Schritt kann beliebig oft ausgeführt werden.
4. Der Benutzer klickt auf den **OK**-Button.
5. OpenCAGE speichert den geänderten Wert zum jeweiligen Attribut.

Analysis

Führt eine Analyse auf einer View aus.

Vorbedingungen

- Es existiert mindestens ein Graph-Window.

mögliche Einstiegspunkte

- Masterbar
- Menüleiste
- Skript

Ablauf

1. OpenCAGE öffnet den *Analysis-Master* (Abschnitt namens *Analysis-Master* in Kapitel 4).
2. Der Benutzer wählt die gewünschte Analyse aus. Der Benutzer drückt den **To Window** Button.
3. OpenCAGE ändert den Mauszeiger in den Analyze-Window- *Pointer* (Abschnitt namens *Pointer* in Kapitel 4).
4. Der Benutzer bewegt den Pointer in eine Graph-Area und klickt die linke Maustaste.
5. OpenCAGE öffnet den Dialog *New User-View* (Abschnitt namens *New User-View* in Kapitel 4).
6. Er vergibt einen Namen für die neue View, die das Ergebnis der Analyse enthält. Der Benutzer startet die Analyse mit dem **Create** Button.
7. OpenCAGE sperrt sämtliche Use-Cases aus dem Bereich RFG Manipulation. Der Dialog *Progress* (Abschnitt namens *Progress* in Kapitel 4) wird angezeigt.
8. Am Ende der Analyse bekommt der Benutzer ein neues Fenster mit der Graph-Window der neuen View. OpenCAGE erlaubt den Zugriff auf die gesperrten Use-Cases wieder.

Alternative Abläufe

erster alternativer Ablauf

Bedingung: Der Benutzer bricht die Analyse mit Hilfe des Dialogs *Progress* (Abschnitt namens *Progress* in Kapitel 4) ab.

1. Der Analysevorgang wird wenn möglich abgebrochen und bleibt wirkungslos. Der Benutzer kann den Vorgang von neuem starten. OpenCAGE erlaubt den Zugriff auf die gesperrten Use-Cases wieder.

zweiter alternativer Ablauf

Bedingung: Der Benutzer benutzt anstelle des **To Window** Buttons den Button **To RFG**. Ab Schritt 5 wird der Use-Case regulär durchgeführt.

1. Die Analyse wird auf dem ganzen RFG durchgeführt.

Collapse

Ein oder mehrere Knoten werden zu einem AC-Knoten zusammengefasst.

Vorbedingungen

- Es ist mindestens ein Knoten im aktuellen Graph-Window selektiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE fragt den Benutzer mit dem Dialog *Collapse* (Abschnitt namens *Collapse*) nach einem Namen für den neuen Knoten.
2. Der Benutzer gibt den neuen Knotennamen für den Knoten ein.
3. OpenCAGE führt auf die selektierten Knoten eine Collapse-Operation aus. Die genaue Spezifikation dieser Operation ist der RFG-Spezifikation zu entnehmen.

Expand

Macht die Funktion Collapse rückgängig. Holt alle Knoten in dem selektierten Knoten eine Hierarchieebene hoch.

Vorbedingungen

Es ist genau ein AC-Knoten im aktuellen Graph-Window selektiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE führt auf die selektierten Knoten eine Expand-Operation aus. Die genaue Spezifikation dieser Operation ist der RFG-Spezifikation zu entnehmen.

View-Master

Die folgenden Use-Cases beschreiben Operationen, die im View-Master auf kompletten Views arbeiten bzw. diese verwalten. Aktionen, die den RFG manipulieren, können nicht angewendet werden, wenn die betreffende View durch den Benutzer explizit oder durch eine gestartete Analyse implizit auf read-only gesetzt wurde. Die Basis-Views sind permanent gesperrt.

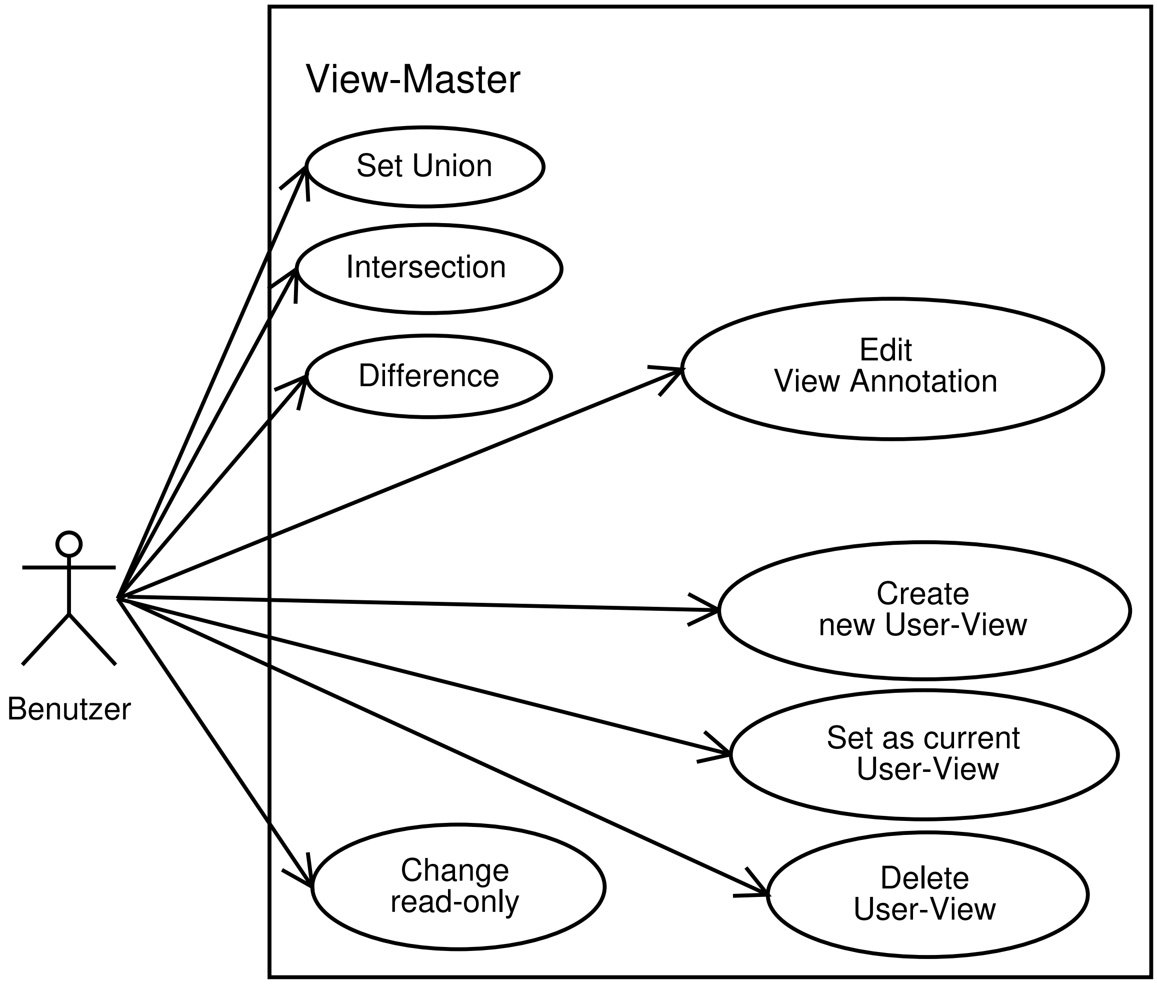


Abbildung 5-4. Use-Case-Diagramm: Viewmaster

Change read-only

Ändert den read-only-Status einer View.

Vorbedingungen

- Es existiert neben den Basis-Views noch mindestens eine weitere View, die nicht gesperrt ist.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. Der Benutzer selektiert im View-Master eine View, die keine Basis-View ist und die nicht gesperrt ist.
2. OpenCAGE invertiert den read-only-Status der View und stellt die selektierte View durch das entsprechende Pixmap dar, siehe *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* (Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* in Kapitel 2)
3. Ist die View nun auf read-only gesetzt, setzt OpenCAGE das Read-only-Statusfeld in allen Graph-Windows dieser View, ansonsten wird das Statusfeld zurückgesetzt.

Create new User-View

Erzeugt eine neue, leere User-View.

Vorbedingungen

Das Fenster des View-Master ist geöffnet.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE zeigt den Dialog *New User-View* (Abschnitt namens *New User-View* in Kapitel 4) an, in dem der Benutzer ein Namen für die neue User-View eingeben kann.
2. Der Benutzer beendet den Dialog erfolgreich durch Drücken des **Rename**-Button, d.h die neue User-View wird erstellt und erhält den eingegebenen Namen.

Delete User-View

Löscht eine User-View.

Vorbedingungen

- Das Fenster des View-Master ist geöffnet.
- Genau eine User-View ist selektiert.
- Die selektierte User-View ist nicht gesperrt.
- Die selektierte UserView ist nicht auf read-only gesetzt.
- Die selektierte User-View ist nicht die aktuelle User-View.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE löscht die selektierte User-View.

Alternative Abläufe

erster alternativer Ablauf

Bedingung: In der selektierten User-View sind noch RFG-Objekte enthalten.

1. OpenCAGE öffnet den Dialog *Warning: View not Empty* (Abschnitt namens *Warning: View not Empty* in Kapitel 4).
2. Klickt der Benutzer auf den **OK**-Button, wird die User-View mit den darin enthaltenen RFG-Objekten gelöscht, ansonsten bleibt die User-View samt ihren enthaltenen RFG-Objekten erhalten.

Set as current User-View

Setzt eine User-View als aktuelle User-View.

Vorbedingungen

- Das Fenster des View-Master ist geöffnet. Es ist eine User-View selektiert, die nicht gesperrt ist.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE zeigt die bisherige aktuelle User-View durch das Pixmap für normale Views an. Die selektierte User-View wird durch das Pixmap für die aktuelle User-View an. Die Pixmaps werden vom Benutzer spezifiziert, siehe *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* (Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* in Kapitel 2).

Intersection

Erzeugt aus der Schnittmenge von zwei Views eine neue View.

Vorbedingungen

- Es sind genau zwei Views im View-Master selektiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE zeigt den Dialog *Intersection* (Abschnitt namens *Intersection* in Kapitel 4) an, welcher die Eingabe des Namens für die neue View verlangt, die beim Schnitt entsteht.
2. Wenn der Dialog mit **Intersect-Button** beendet wird, schneidet OpenCAGE die beiden Views unter Berücksichtigung des *Modify Affinity Threshold*, (Abschnitt namens *Modify Affinity Threshold*).
3. OpenCAGE erzeugt eine neue View, fügt das Ergebnis der Operation in diese View ein und zeigt die View in einem neuen View-Window an.

Set Union

Erzeugt aus der Vereinigung von zwei Views eine neue View.

Vorbedingungen

- Es sind genau zwei Views im View-Master selektiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE zeigt den Dialog *Set Union* (Abschnitt namens *Set Union* in Kapitel 4) an, in dem der Benutzer den Namen der zu erzeugenden Schnitt-View eingeben kann.
2. Wenn der Dialog mit dem **Unite**-Button verlassen wird, vereinigt OpenCAGE die beiden Views zu einer neuen View. Dabei wird mit Hilfe des Affinity Threshold (siehe *Modify Affinity Threshold*, Abschnitt namens *Modify Affinity Threshold*) bestimmt, ob zwei AC-Knoten als gleich anzusehen sind.
3. OpenCAGE erzeugt eine neue View, fügt die Ergebnisse der Operation in diese View ein und zeigt die View in einem neuen Graph-Window an.

Difference

Erstellt die Differenzmenge von zwei Views in einer neuen View.

Vorbedingungen

- Es ist genau eine View im View-Master selektiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE öffnet den Dialog *Select one View* (Abschnitt namens *Select one View* in Kapitel 4).
2. Der Benutzer drückt den **OK**-Button.
3. OpenCAGE zeigt den Dialog *Difference* (Abschnitt namens *Intersection* in Kapitel 4) an, welcher die Eingabe des Namens für die neue View verlangt, die bei der Differenzmengenbildung entsteht.

4. OpenCAGE bildet die Differenzmenge beider Views, wobei die im View-Master selektierte View die Minuendmenge darstellt. Die im Dialog *Select one View* selektierte stellt die Subtrahendmenge dar. Dabei wird mit Hilfe des Affinity Threshold (siehe auch *Modify Affinity Threshold*, Abschnitt namens *Modify Affinity Threshold*) bestimmt, ob zwei AC-Knoten als gleich anzusehen sind.
5. OpenCAGE erzeugt eine neue View, fügt die Ergebnisse der Operation in diese View ein und zeigt die View in einem neuen Graph-Window an.

Edit View Annotation

Zeigt im Dialog *Edit View Annotation* (Abschnitt namens *Edit View Annotation* in Kapitel 4) die Beschreibung der View an.

Vorbedingungen

- Mindestens eine View existiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE öffnet den Dialog *Edit View Annotation* (Abschnitt namens *Edit View Annotation* in Kapitel 4)
2. Der Benutzer kann den Beschreibungstext ändern, falls bereits Text vorhanden ist. Ansonsten ist das Textfeld noch leer und der Benutzer kann Text einfügen.
3. Der Benutzer bestätigt seine Eingabe durch Drücken des **OK**-Button.
4. OpenCAGE speichert die neue Beschreibung zur View.

Alternative Abläufe

erster alternativer Ablauf

Bedingung: Die View, deren Annotation geändert werden soll, ist gesperrt.

1. Das Texteingabefeld im Dialog *Edit View Annotation* (Abschnitt namens *Edit View Annotation* in Kapitel 4) ist ausgegraut, so dass der Benutzer den Text nicht ändern kann.
2. Der Benutzer bestätigt seine Eingabe durch Drücken des **OK**-Button.

Window-Master

Die folgenden Use-Cases beschreiben Operationen, die die Verwaltung der Fenster innerhalb des Window-Masters steuern.

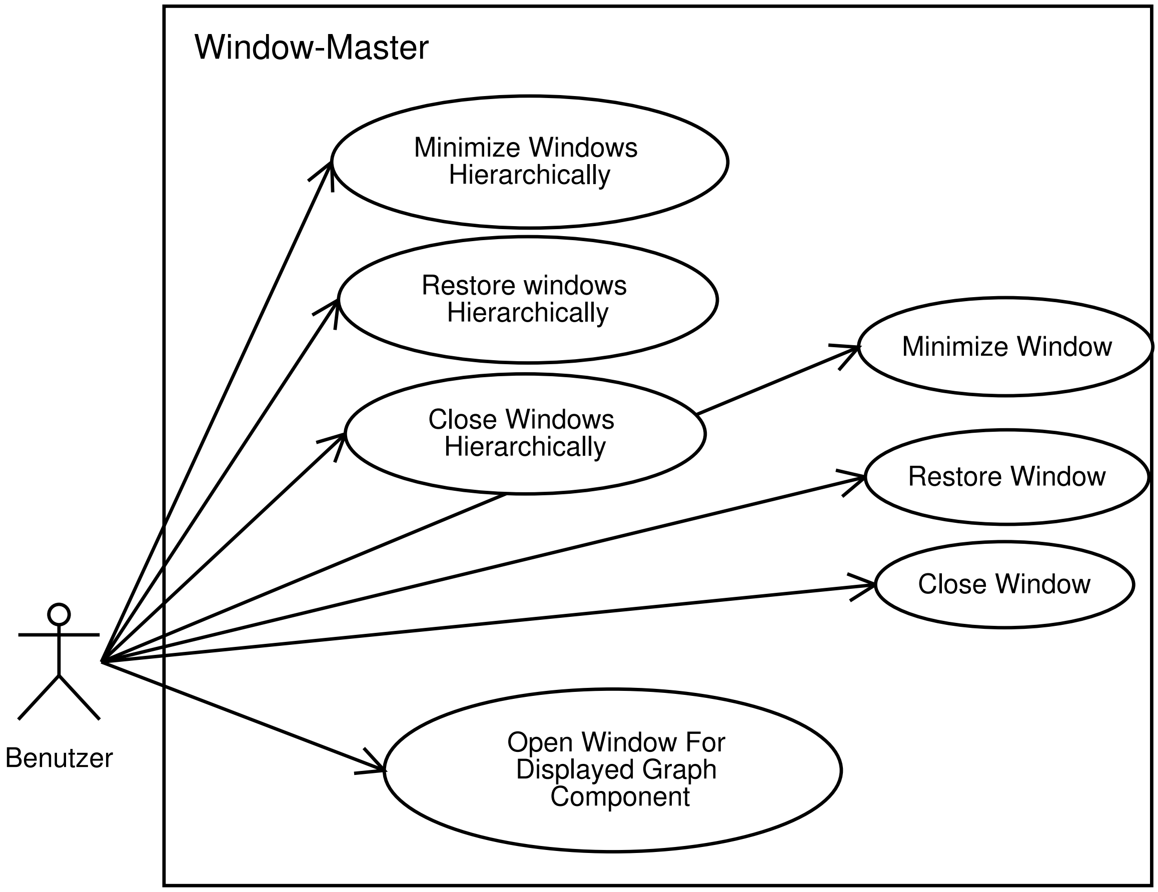


Abbildung 5-5. Use-Case-Diagramm: Window-Master

Close Window

Schließt das im Window-Master selektierte Graph-Window.

Vorbedingungen

- Ein Fenstereintrag im Window-Master ist selektiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE schließt das Graph-Window, das der im Window-Master selektierte Fenstereintrag referenziert, und entfernt den selektierten Fenstereintrag.

Close Windows Hierarchically

Schließt die Graph-Windows aller hierarchisch tiefer liegenden Fenstereinträge unter einem im Window-Master selektierten.

Vorbedingungen

- Ein Grapheneintrag im Window-Master ist selektiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE schließt sämtliche Graph-Windows, die von Fenstereinträgen im Window-Master referenziert werden, welche sich hierarchisch unter dem selektierten Grapheneintrag im Window-Master befinden, und das Fenster, das durch den selektierten Grapheintrag repräsentiert wird.
2. OpenCAGE entfernt sämtliche Fenstereinträge aus dem Window-Master, die sich hierarchisch unter dem selektierten Fenstereintrag befinden.

Minimize Window

Minimiert das im Window-Master selektierte Graph-Window.

Vorbedingungen

- Ein Fenstereintrag im Window-Master ist selektiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE minimiert das im Window-Master selektierte Graph-Window. Wenn das Graph-Window bereits minimiert ist, geschieht nichts.

Minimize Windows Hierarchically

Minimiert alle Fenster in der Hierarchie unter dem selektierten Grapheneintrag.

Vorbedingungen

- Ein Grapheneintrag im Window-Master ist selektiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE minimiert alle Graph-Windows, die in der Hierarchie unter dem selektierten Grapheneintrag liegen, und das Fenster, das durch den selektierten Grapheintrag repräsentiert wird.Graph-Windows, die bereits minimiert sind, bleiben unverändert.

Restore Window

Bringt das im Window-Master selektierte Graph-Window auf die ursprüngliche Größe.

Vorbedingungen

- Ein Fenstereintrag im Window-Master ist selektiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE bringt das im Window-Master selektierte Graph-Window auf die Größe, die es vor der Minimierung hatte und plaziert es an die Stelle, an der es vor der Minimierung war. Ist das Graph-Window nicht minimiert, bleibt es unverändert.

Restore Windows Hierarchically

Bringt die im Window-Master hierarchisch unter dem selektierten Grapheneintrag liegenden Fenster auf ursprüngliche Größe.

Vorbedingungen

- Ein Grapheneintrag im Window-Master ist selektiert.

mögliche Einstiegspunkte

- Kontextmenü
- Skript

Ablauf

1. OpenCAGE bringt sämtliche Graph-Windows, die von Fenstereinträgen im Window-Master referenziert werden, welche sich hierachisch unter dem selektierten Grapheneintrag im Window-Master befinden, auf die Größe und an die Position, die sie vor der Minimierung hatten. Dasselbe geschieht mit dem Fenster, das durch den markierten Grapheneintrag repräsentiert wird. Graph-Windows, die nicht minimiert sind, bleiben unverändert.

Open Window For Displayed Graph Component

Erstellt ein neues Graph-Window, das den Graphen zum selektierten Grapheneintrag im Window-Master zeigt.

Vorbedingungen

- Ein Grapheneintrag im Window-Master ist selektiert.

mögliche Einstiegspunkte

- Kontextmenü

- Skript

Ablauf

1. OpenCAGE öffnet ein neues Graph-Window, in dem die Graphenkomponente gezeigt wird, die der selektierte Grapheneintrag im Window-Master referenziert. Anschließend fügt OpenCAGE auf der gleichen Ebene wie der selektierte Grapheneintrag einen neuen Grapheneintrag im Window-Master ein, der das neu erstellte Graph-Window referenziert.

Workbench

Das folgende Diagramm beschreibt die Use-Cases, die in der Workbench zur Verfügung stehen. Das sind im wesentlichen Use-Cases zur Verwaltung des RFGs, das Hilfesystem und die Use-Cases für Clipboard-View und Trashcan-View.

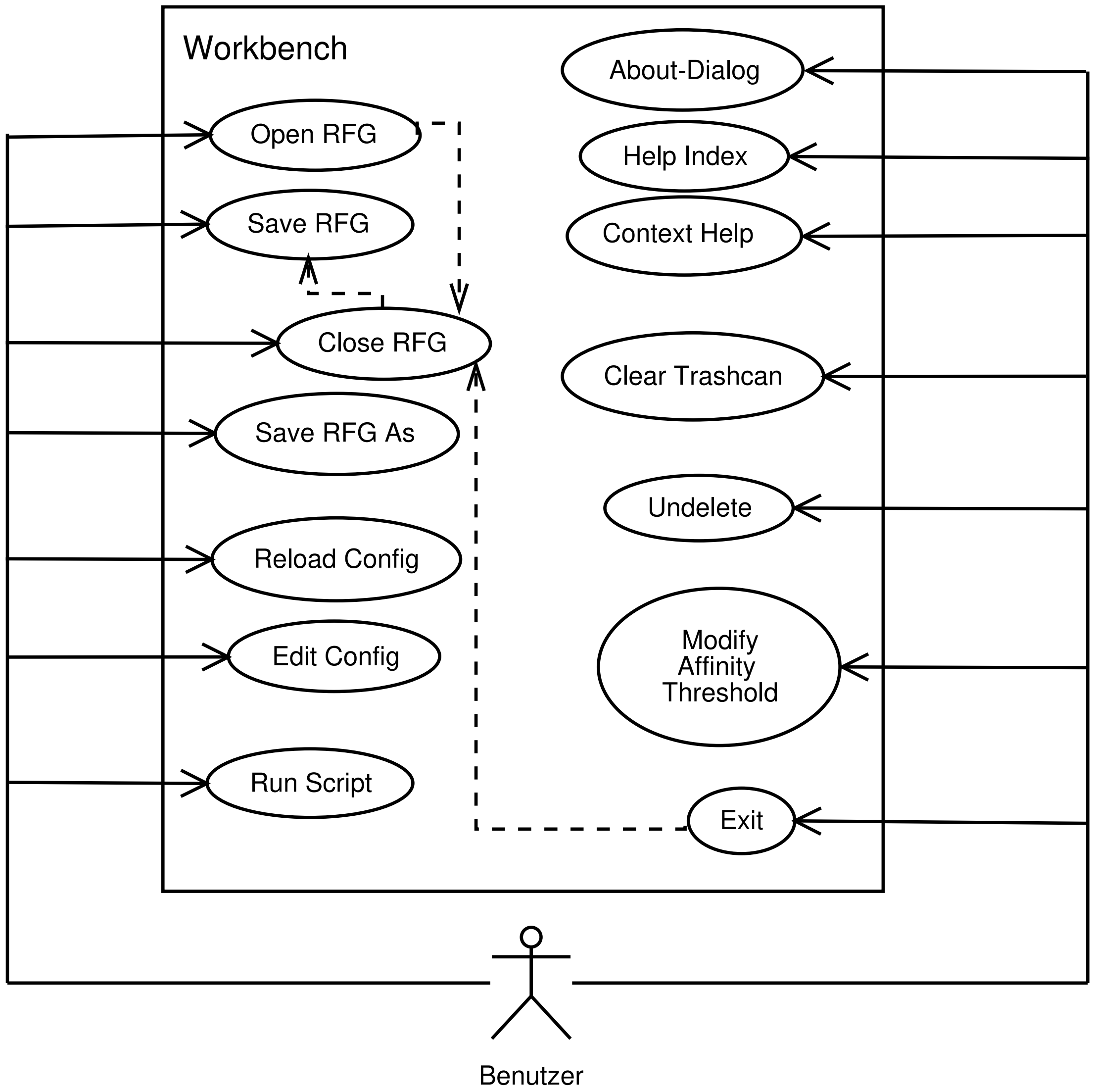


Abbildung 5-6. Use-Case-Diagramm: Workbench

About-Dialog

Zeigt das Programm-Info-Fenster an.

Vorbedingungen

Keine

mögliche Einstiegspunkte

- Menüleiste

Ablauf

1. OpenCAGE öffnet den Dialog *About* (Abschnitt namens *About* in Kapitel 4).

Clear Trashcan

Löscht den Inhalt der Trashcan-View.

Vorbedingungen

Keine

mögliche Einstiegspunkte

- Menüleiste
- Skript

Ablauf

1. OpenCAGE entfernt alle RFG-Objekte aus der Trashcan-View.

Close RFG

Schließt den RFG.

Vorbedingungen

Keine

mögliche Einstiegspunkte

- Menüleiste
- Skript
- durch Use-Case *Open RFG* (Abschnitt namens *Open RFG*)
- durch Use-Case *Exit* (Abschnitt namens *Exit*)

Ablauf

1. OpenCAGE führt den Use-Case *Save RFG* (Abschnitt namens *Save RFG*) unter Nutzung des Dialogs *Unsaved Changes* (Abschnitt namens *Unsaved Changes* in Kapitel 4) aus.

2. OpenCAGE schließt alle Graph-Windows, den View-Master und den Window-Master des dargestellten RFGs.

Alternative Abläufe

erster alternativer Ablauf

Bedingung: Der Dialog *Select File* (Abschnitt namens *Select File* in Kapitel 4) wurde abgebrochen.

1. OpenCAGE bricht die Aktion *Close RFG* ab.

Edit Config

Lädt den Editor mit der Programmkonfigurationsdatei.

Vorbedingungen

Keine

mögliche Einstiegspunkte

- Menüleiste

Ablauf

1. OpenCAGE öffnet die benutzerdefinierte Konfigurationsdatei (Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* in Kapitel 2) mit dem Editor.

Alternative Abläufe

erster alternativer Ablauf

Bedingung: Es existiert keine programmspezifische Konfigurationsdatei.

1. OpenCAGE öffnet die globale Konfigurationsdatei (Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* in Kapitel 2) mit dem Editor und speichert den Inhalt nach der Änderung in der benutzerdefinierten Konfigurationsdatei.

zweiter alternativer Ablauf

Bedingung: Der Editor wird nicht gefunden.

1. OpenCAGE öffnet den Dialog *Error: Editor not found* (Abschnitt namens *Error: Editor not found* in Kapitel 4).

Exit

Beendet das Programm.

Vorbedingungen

Keine

mögliche Einstiegspunkte

- Menüleiste
- Skript

Ablauf

1. Wenn eine RFG-Datei geöffnet ist, wird der Use-Case *Close RFG* (Abschnitt namens *Close RFG*) gestartet. Dann beendet sich OpenCAGE. Wenn keine RFG-Datei geöffnet ist, beendet sich OpenCAGE sofort.

Alternative Abläufe

erster alternativer Ablauf

Bedingung: Der Dialog *Select File* (Abschnitt namens *Select File* in Kapitel 4) wurde abgebrochen.

1. OpenCAGE bricht die Aktion *Exit* ab.

Context Help

Zeigt die Hilfeseite an, die zum aktuellen Programm-Kontext passt.

Vorbedingungen

Keine

mögliche Einstiegspunkte

- Menüleiste
- Help-Button

Ablauf

1. OpenCAGE startet den Browser und zeigt die Hilfeseite an, die zum aktuellen Programm-Kontext passt.

Alternative Abläufe

erster alternativer Ablauf

Bedingung: Der angegebene Browser existiert nicht.

1. OpenCAGE zeigt die Fehlermeldung, dass der Browser nicht vorhanden ist (Dialog *Error: Browser not found* (Abschnitt namens *Error: Browser not found* in Kapitel 4)).

zweiter alternativer Ablauf

Bedingung: Es ist bereits ein Browser-Fenster geöffnet.

1. OpenCAGE zeigt die Hilfeseite in einem dieser Browser-Fenster an.

Help Index

Zeigt den Index der Hilfe an.

Vorbedingungen

Keine

mögliche Einstiegspunkte

- Menüleiste

Ablauf

1. OpenCAGE startet den Browser und zeigt den Hilfeindex an.

Alternative Abläufe

erster alternativer Ablauf

Bedingung: Der angegebene Browser existiert nicht.

1. OpenCAGE zeigt die Fehlermeldung, dass der Browser nicht vorhanden ist (Dialog *Error: Browser not found* (Abschnitt namens *Error: Browser not found* in Kapitel 4)).

zweiter alternativer Ablauf

Bedingung: Es ist bereits ein Browser-Fenster geöffnet.

1. OpenCAGE zeigt die Hilfeseite in einem dieser Browser-Fenster an.

Open RFG

Öffnet einen neuen RFG.

Vorbedingungen

Keine

mögliche Einstiegspunkte

- Menüleiste
- Skript

Ablauf

1. OpenCAGE überprüft, ob kein RFG geöffnet ist.
2. OpenCAGE öffnet den Dialog *Select File* (Abschnitt namens *Select File* in Kapitel 4), in welchem der Benutzer den RFG auswählen kann.
3. OpenCAGE lädt den RFG, erstellt die Basis-Views und lädt die projektspezifischen Konfigurationsdaten

Alternative Abläufe

erster alternativer Ablauf

Bedingung: Es ist ein RFG geöffnet.

1. OpenCAGE startet den Use-Case *Close RFG* (Abschnitt namens *Close RFG*); anschließend wird der reguläre Ablauf des Use-Cases wiederaufgenommen.

zweiter alternativer Ablauf

Bedingung: OpenCAGE kann keine projektspezifischen Daten finden.

1. OpenCAGE verwendet Standarddaten. Es werden nur die Views im View-Master dargestellt und keine View-Windows erzeugt.

dritter alternativer Ablauf

Bedingung: Der Dialog *Select File* (Abschnitt namens *Select File* in Kapitel 4) wurde abgebrochen.

1. OpenCAGE bricht die Aktion *Open RFG* ab.

Reload Config

Lädt die Programmparameter neu.

Vorbedingungen

Keine

mögliche Einstiegspunkte

- Menüleiste

Ablauf

1. OpenCAGE überprüft, ob die Programmkonfigurationsdatei existiert.
2. OpenCAGE lädt die Programmparameter neu.
3. OpenCAGE passt Fenster und Fensterinhalte entsprechend an die neuen Programmparameter an.

Alternative Abläufe

erster alternativer Ablauf

Bedingung: Es existiert keine programmspezifische Konfigurationsdatei.

1. OpenCAGE lässt die Programmparameter unverändert.

Run Script

Lädt und führt ein Skript aus.

Vorbedingungen

- Der Benutzer hat die RFG-Datei geöffnet, auf der er das Skript ablaufen lassen will.

mögliche Einstiegspunkte

- Menüleiste

Ablauf

1. OpenCAGE zeigt den Dialog *Select File* (Abschnitt namens *Select File* in Kapitel 4) an.
2. Der Benutzer wählt die Datei mit dem Skript zu der RFG-Datei aus.
3. OpenCAGE führt das Skript auf der RFG-Datei aus. Dabei wird nur der Dialog *Progress* (Abschnitt namens *Progress* in Kapitel 4) angezeigt.
4. Am Ende des Skripts wird das Resultat, die neu entstandene RFG-Datei angezeigt. Der Benutzer kann mit OpenCAGE normal weiterarbeiten.

Alternative Abläufe

erster alternativer Ablauf

Bedingung: Das Skript enthält einen Fehler oder eine nicht erlaubte bzw. nicht mögliche Funktion.

1. OpenCAGE führt analog zum regulären Ablauf das Skript aus. Sobald der Fehler auftritt, wird der Ablauf des Skripts gestoppt. Es wird mit einer Fehlermeldung die Position des Ablaufs des Skripts wird auf der Error-Konsole (siehe *Error-Log-Console*,Abschnitt namens *Error-Log-Console* in Kapitel 4) ausgegeben.

Save RFG

Speichert den RFG unter seinem aktuellen Namen.

Vorbedingungen

- Keine

mögliche Einstiegspunkte

- Menüleiste
- Skript
- durch Use-Case *Close RFG* (Abschnitt namens *Close RFG*)
- durch Use-Case *Exit* (Abschnitt namens *Exit*) oder *Exit* (Abschnitt namens *Open RFG*) über Use-Case *Close RFG* (Abschnitt namens *Close RFG*)

Ablauf

1. OpenCAGE überprüft, ob eine Datei mit dem aktuellen Namen existiert.
2. OpenCAGE speichert den RFG unter seinem aktuellen Namen und die zugehörigen projektspezifischen Daten in der entsprechenden Datei (siehe *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* in Kapitel 2) ab.

Alternative Abläufe

erster alternativer Ablauf

Bedingung: Es ist keine Backupeinstellung gesetzt oder es existiert noch keine Datei mit dem aktuellen Namen.

1. OpenCAGE legt keine Backup-Dateien an und fährt mit dem weiteren Ablauf fort.

Save RFG as

Speichert den RFG.

Vorbedingungen

Keine

mögliche Einstiegspunkte

- Menüleiste
- Skript

Ablauf

1. OpenCAGE öffnet den Dialog *Select File* (Abschnitt namens *Select File* in Kapitel 4), in dem der Benutzer den Dateinamen für die RFG-Datei auswählt.
2. OpenCAGE speichert den RFG unter den neuen Namen und die zugehörigen projektspezifischen Daten in einer weiteren entsprechenden Datei (siehe *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* in Kapitel 2) ab.

Undelete

Macht die letzte Lösch-Aktion rückgängig.

Vorbedingungen

- Mindenstens ein Knoten befindet sich in der Trashcan-View.
- Es wurden keine RFG-Manipulationsoperation (Abschnitt namens *RFG manipulation*) seit der letzten Aktion auf die *Trashcan-View* (Abschnitt namens *Trashcan-View* in Kapitel 2) ausgeführt.
- Es wurden keine RFG-Manipulationsoperation (Abschnitt namens *RFG manipulation*) auf die View, zu der die RFG-Objekte gehören, die sich gerade in der *Trashcan-View* (Abschnitt namens *Trashcan-View* in Kapitel 2)befinden, ausgeführt.

mögliche Einstiegspunkte

- Menüleiste
- Skript

Ablauf

1. Alle Knoten und induzierten Kanten werden in die ursprüngliche View verschoben. Sie werden aus der Trashcan-View entfernt.

Modify Affinity Threshold

Verändert die Einstellungen des Affinity Threshold.

Vorbedingungen

Keine

mögliche Einstiegspunkte

- Menüleiste
- Skript

Ablauf

1. OpenCAGE öffnet den Dialog *Affinity Threshold* (Abschnitt namens *Affinity Threshold* in Kapitel 4).
2. Der Benutzer ändert die Einstellungen.
3. OpenCAGE übernimmt die Einstellungen.

Anhang A. Changes

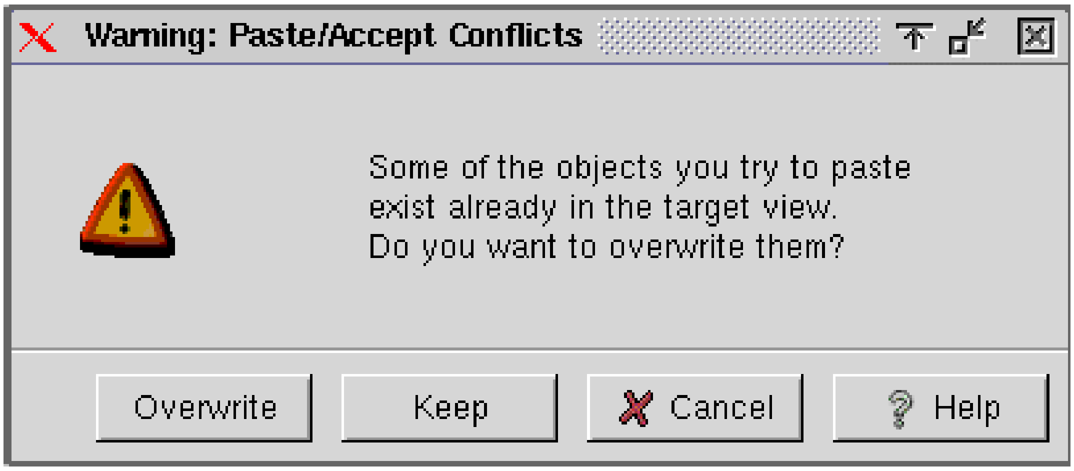
In diesem Kapitel werden nachträgliche Veränderungen, die z.B. nach einem Review oder einer Anforderungsänderung notwendig werden, dargestellt. Diese Veränderungen werden nach jeder Iterationsstufe in die Spezifikation eingearbeitet. Vorteil dieser Vorgehensweise ist, den Beteiligten einen besseren Umgang mit der Spezifikation zu ermöglichen. Innerhalb der Iterationsphasen sind sie mit der zu Beginn der Iterationsphase bestehenden Spezifikation vertraut und können sich somit schnell auf den neuesten Stand bringen. Dennoch wird die Konsistenz der Spezifikation auch auf lange Sicht gewahrt.

Die Referenzen zeigen auf die relevanten Stellen der Spezifikation, die durch die Änderungen betroffen sind.

Änderungen nach dem 2001-02-15 (2. externes Review)

Paste

Paste-Konflikte werden für Knoten und Kanten in gleicher Weise behandelt. Siehe Änderungen im Dialog *Warning: Paste Conflict* (Abschnitt namens *Warning: Paste Conflict* in Kapitel 4). Daher wurden die Buttons umbenannt in **Overwrite** und **Keep**.



Dieser Dialog zeigt einen Konflikt beim Einfügen von RFG-Objekten an.

Abbildung A-1. Dialog: Warning: Paste Conflict

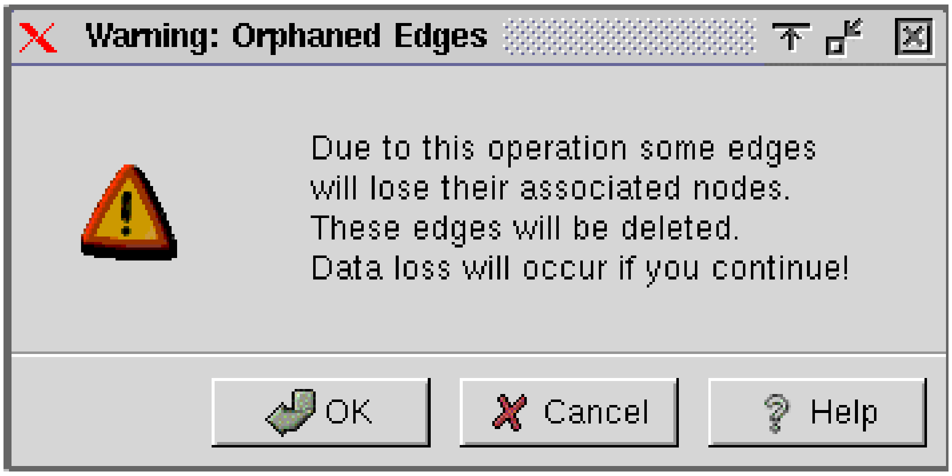
Accept

Akzeptieren ist auf Knoten und Kanten gleichwertig anwendbar.

OpenCAGE überprüft, ob durch die Accept-Aktion Orphaned Edges entstehen. In solch einem Fall zeigt OpenCAGE den *Dialog: Orphaned Edges* (Abbildung A-2) an. Drückt der Benutzer den Ok-Button werden die Kanten gelöscht und die Accept-Aktion durchgeführt. Drückt der Benutzer den Cancel-Button, wird die Accept-Aktion abgebrochen und die betroffenen Kanten werden durch Conflict-Highlighting hervorgehoben.

Konflikte beim Akzeptieren werden wie Konflikte bei Paste behandelt, vergleiche *Paste* (Abschnitt namens *Paste* in Kapitel 5). Der Benutzer erhält im Dialog *Warning: Paste Conflict* (Abschnitt namens *Warning: Paste Conflict* in Kapitel 4) die Auswahl, ob entsprechende Knoten mit den Attributen des zu akzeptierenden Knoten überschrieben werden sollen, oder die alten Attribute beibehalten werden. Analog gilt dies auch für Kanten. Die View enthält nach dem Akzeptieren die Vereinigung der Menge der RFG-Objekte, die vor dem Akzeptieren in der aktuellen User-View vorhanden waren, mit den zu akzeptierenden RFG-Objekten .

Die zweite Vorbedingung (es dürfen nur Kanten selektiert sein, deren Start- und Endknoten selektiert sind) im Use-Case *Accept* (Abschnitt namens *Accept* in Kapitel 5) entfällt.

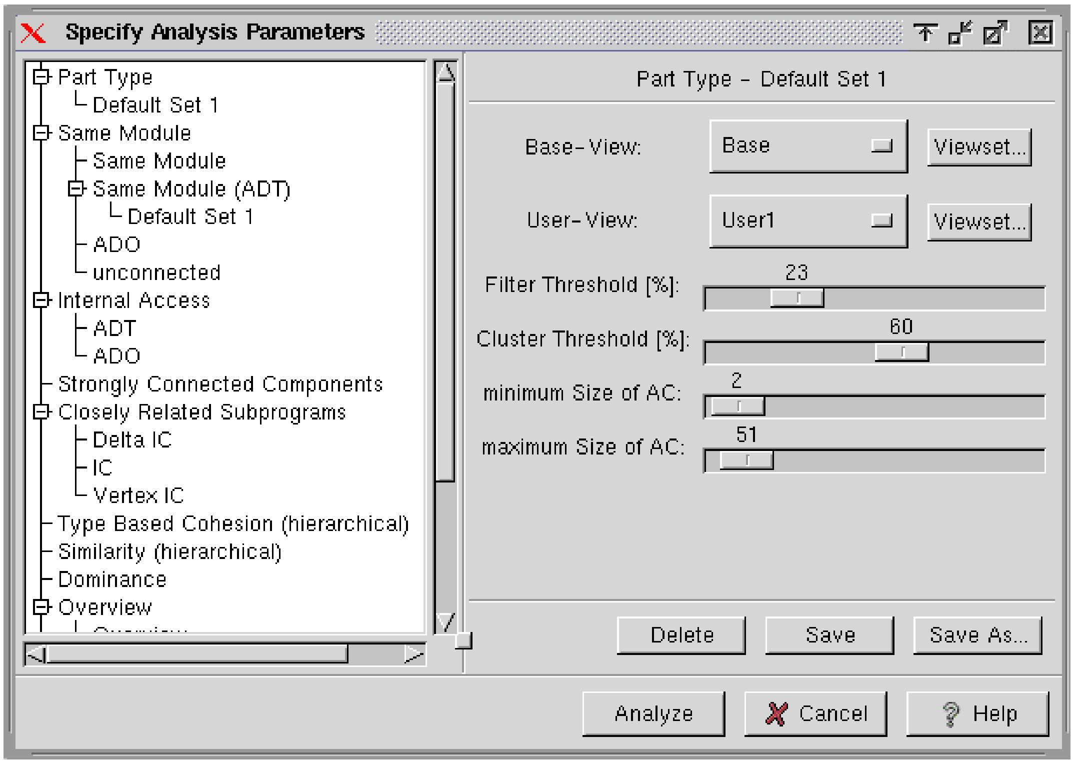


Dieser Dialog wird angezeigt, wenn bei einer Aktion verwaiste Kanten entstanden sind.

Abbildung A-2. Dialog: Orphaned Edges

Analyze

Analysen können Views und Viewmengen als Parameter verlangen. Die Analyse spezifiziert, ob sie Views oder Viewmengen als Parameter verlangt und wieviele solcher Parameter sie verlangt. Als Folge davon ändern sich der Screenshot des Analysis-Master, die Beschreibung des Analysis-Master und die Beschreibung der Mode-Master in den GUI-Konzepten. Vergleiche *Mode-Master* (Abschnitt namens *Mode-Master* in Kapitel 3).



In diesem Dialog können die Analyseparameter verändert und Analysen gestartet werden.

Abbildung A-3. Dialog: Analysis-Master

Änderungen im Dialog *Analysis-Master* (Abschnitt namens *Analysis-Master* in Kapitel 4):

Der **To Window**-Button ist umbenannt in **Analyze**. Dieser Button startet (ohne Fensterauswahl) die Analyse. Views werden nun bei den Analyseparametern in der rechten Fensterhälfte ausgewählt.

Somit fällt der Analyze-Window-Pointer weg, vergleiche *Pointer* (Abschnitt namens *Pointer* in Kapitel 4).

Der Benutzer kann mit Hilfe der Drop-Down-Box eine einzelne View auswählen. Will er eine Viewmenge spezifizieren, dann drückt er den „Viewset...“-Button. Wenn eine Analyse eine einzige View als diesen Parameter zulässt, wird dieser Button ausgegraut. Dadurch öffnet OpenCAGE den Dialog *Select Views* (Abschnitt namens *Select Views* in Kapitel 4). Hier kann der Benutzer eine Menge von Views auswählen. Wenn eine Analyse nur eine einzige View als diesen Parameter zulässt, wird dieser Button ausgegraut.

Backup-Datei

OpenCAGE besitzt eine Backup-Funktion, die als programmspezifische Einstellung aktiviert wird, vergleiche *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* (Abschnitt namens *Programm-, Projektspezifische Einstellungen und Konfigurationsdateien* in Kapitel 2). Beim Aktivieren dieser Funktion, überprüft OpenCAGE vor jedem Speichern, ob bereits eine Datei mit gleichem Namen existiert. Ist dies der Fall, so wird sie durch Anhängen der Endung „.bak“ umbenannt (Existiert beriets eine Datei mit der Endung „.bak“, wird sie einfach überschrieben). Danach speichert OpenCAGE seine eigene Datei (siehe *Übersicht über Aktionen, die auf Backup-Funktionalität zugreifen*) unter dem ursprünglichen Namen ab.

Tabelle A-1. Übersicht über Aktionen, die auf Backup-Funktionalität zugreifen

Dateityp	Use-Cases
RFG/projektspezifische Einstellungen	• *Save RFG* (Abschnitt namens *Save RFG* in Kapitel 5) • *Save RFG as* (Abschnitt namens *Save RFG as* in Kapitel 5)

Kontextmenüs

Kontextmenüs werden objektabhängig GUI-Einheiten übergreifend (Graph-Window/Viewmaster) erzeugt, vergleiche *Kontextmenüs* (Abschnitt namens *Kontextmenüs* in Kapitel 3). Es existieren Kontextmenüs für:

- Composite-Knoten
- Entitätsknoten
- Kanten
- Views
- Fenstereinträge des Window-Masters
- Grapheneinträge des Window-Masters

Sind mehrere Objekte unterschiedlichen Typs selektiert, so sind nur die Menüeinträge enthalten, die in den Kontextmenüs aller Objekttypen der selektierten Objekte dargestellt sind. Ist eine Aktion nur dann ausführbar, wenn eine bestimmte Anzahl von Objekten selektiert ist, so ist der entsprechende Menüeintrag ausgegraut. Sind alle Untermenüeinträge eines Kontextmenüs ausgegraut, so ist der entsprechende Menüeintrag, der sich eine Menüebene überhalb dieser Einträge befindet, ebenfalls ausgegraut.

View-Master und Graph-Windows stellen nach wie vor bezüglich dem Selektieren zwei verschiedene Einheiten dar. *Ausnahme:* Klickt der Benutzer auf den Hintergrund einer Graph-Area, der einer View zugeordnet ist, dann wird sowohl die Selektion im View-Master als auch die im Graph-Window aufgehoben. Außerdem wird im View-Master die View selektiert, die dem Hintergrund der Graph-Area zugeordnet ist. Vergleiche *Selektieren* (Abschnitt namens *Selektieren* in Kapitel 3).

Das graphische Objekt, welches im View-Master eine View darstellt, besitzt also das gleiche Kontextmenü wie das Graph-Window, welches eine ganze View referenziert, nämlich das Kontextmenü View. Im Kontextmenü des Graph-Windows, welches eine ganze View referenziert, ist der Eintrag **Make** ausgegraut, da alle Aktionen, die sich unter diesen Menüeintrag befinden (Set Union, Intersection, Difference), als Vorbedingung haben, dass genau zwei Views selektiert sind. Sind in einem Graph-Window ein Entitätsknoten und eine Kante selektiert, befindet sich der Eintrag **Rename** nicht im entsprechenden Kontextmenü, da er sich nicht im Kontextmenü für Kanten befindet.

Composite-Knoten

- RFG

  RFG⟶Accept: *Accept* (Abschnitt namens *Accept* in Kapitel 5)

  RFG⟶Accept as: *Accept as* (Abschnitt namens *Accept as* in Kapitel 5)

  RFG⟶Reject: *Reject* (Abschnitt namens *Reject* in Kapitel 5)

  RFG⟶Collapse: *Collapse* (Abschnitt namens *Collapse* in Kapitel 5)

  RFG⟶Expand: *Expand* (Abschnitt namens *Expand* in Kapitel 5)

- Edit

  Edit⟶Copy: *Copy* (Abschnitt namens *Copy* in Kapitel 5)

  Edit⟶Cut: *Cut* (Abschnitt namens *Cut* in Kapitel 5)

  Edit⟶Paste: *Paste* (Abschnitt namens *Paste* in Kapitel 5)

- Attributes

  Attributes⟶Show: *Show attributes* (Abschnitt namens *Show attributes* in Kapitel 5)

  Attributes⟶Hide: *Hide attributes* (Abschnitt namens *Hide attributes* in Kapitel 5)

  Attributes⟶Edit: *Edit Attributes* (Abschnitt namens *Edit Attributes* in Kapitel 5)

- Conflict-Highl

  Conflict-Highl⟶Remove by Sel: *Remove Conflict-Highlighting to Selection* (Abschnitt namens *Remove Conflict-Highlighting to Selection* in Kapitel 5)

  Conflict-Highl⟶Remove all: *Remove Conflict-Highlighting Everywhere* (Abschnitt namens *Remove Conflict-Highlighting Everywhere* in Kapitel 5)

- **Annotate**: *Edit Node Annotation* (Abschnitt namens *Edit Node Annotation* in Kapitel 5)
- **Rename**: *Rename Node* (Abschnitt namens *Rename Node* in Kapitel 5)

vergleiche mit *Graph-Window* (Abschnitt namens *Graph-Window* in Kapitel 4)

Entitätsknoten

- RFG

  RFG⟶Accept: *Accept* (Abschnitt namens *Accept* in Kapitel 5)

  RFG⟶Accept as: *Accept as* (Abschnitt namens *Accept as* in Kapitel 5)

  RFG⟶Reject: *Reject* (Abschnitt namens *Reject* in Kapitel 5)

  RFG⟶Collapse: *Collapse* (Abschnitt namens *Collapse* in Kapitel 5)

- **Show Source**: *Open Source-Code* (Abschnitt namens *Open Source-Code* in Kapitel 5)

- Edit

  Edit⟶Copy: *Copy* (Abschnitt namens *Copy* in Kapitel 5)

  Edit⟶Cut: *Cut* (Abschnitt namens *Cut* in Kapitel 5)

  Edit⟶Paste: *Paste* (Abschnitt namens *Paste* in Kapitel 5)

- Attributes

  Attributes⟶Show: *Show attributes* (Abschnitt namens *Show attributes* in Kapitel 5)

  Attributes⟶Hide: *Hide attributes* (Abschnitt namens *Hide attributes* in Kapitel 5)

  Attributes⟶Edit: *Edit Attributes* (Abschnitt namens *Edit Attributes* in Kapitel 5)

- Conflict-Highl

  Conflict-Highl⟶Remove by Sel: *Remove Conflict-Highlighting to Selection* (Abschnitt namens *Remove Conflict-Highlighting to Selection* in Kapitel 5)

  Conflict-Highl⟶Remove all: *Remove Conflict-Highlighting Everywhere* (Abschnitt namens *Remove Conflict-Highlighting Everywhere* in Kapitel 5)

- **Annotate**: *Edit Node Annotation* (Abschnitt namens *Edit Node Annotation* in Kapitel 5)
- **Rename**: *Rename Node* (Abschnitt namens *Rename Node* in Kapitel 5)

vergleiche mit *Graph-Window* (Abschnitt namens *Graph-Window* in Kapitel 4)

Kante

- RFG

  RFG⟶Accept: *Accept* (Abschnitt namens *Accept* in Kapitel 5)

  RFG⟶Reject: *Reject* (Abschnitt namens *Reject* in Kapitel 5)

- Attributes

  Attributes⟶Show: *Show attributes* (Abschnitt namens *Show attributes* in Kapitel 5)

  Attributes⟶Hide: *Hide attributes* (Abschnitt namens *Hide attributes* in Kapitel 5)

  Attributes⟶Edit: *Edit Attributes* (Abschnitt namens *Edit Attributes* in Kapitel 5)

- Conflict-Highl

  Conflict-Highl⟶Remove by Sel: *Remove Conflict-Highlighting to Selection* (Abschnitt namens *Remove Conflict-Highlighting to Selection* in Kapitel 5)

  Conflict-Highl⟶Remove all: *Remove Conflict-Highlighting Everywhere* (Abschnitt namens *Remove Conflict-Highlighting Everywhere* in Kapitel 5)

- **Show/Hide**: *Show/Hide edges of other views* (Abschnitt namens *Show/Hide edges of other views* in Kapitel 5)

vergleiche mit *Graph-Window* (Abschnitt namens *Graph-Window* in Kapitel 4)

View

gleicher Aufbau wie in *View-Master* (Abschnitt namens *View-Master* in Kapitel 4)

Fenster- und Grapheneinträge im Window-Master

unverändert zu *Window-Master* (Abschnitt namens *Window-Master* in Kapitel 4)

Anhang B. Begriffslexikon

Accept

(engl.)*Accept*

Verschieben von selektierten (incl. der hierarchisch darunter liegenden) *RFG-Objekte* in die aktuelle *User-View*.

Affinity Threshold

(engl.) *Affinity Threshold*

Ein Wert, mit dessen Hilfe überprüft wird, ob zwei *Knoten* als identisch angesehen werden können.

Aktion

(engl.) *Action*

Durch einen *Use-Case* spezifizierter Programmablauf.

Analyse

(engl.) *Analysis*

Eine Analyse als Programmfunktion versucht, Archititekturinformationen über ein *System* aus dem *RFG* zu gewinnen. Diese Informationen werden in einer neuen *View* im RFG gespeichert.

Anforderungsanalyse

(engl.) *Product Engineering*

Bezeichnet den Vorgang des Herausfindens der Anforderungen des Kunden an OpenCAGE

Annotation

(engl.) *Annotation*

Optionaler Beschreibungstext zu Elementen von OpenCAGE (*Kante*, *Knoten*, *View*), der vom Benutzer erstellt und bearbeitet werden kann.

Architektur-Komponente

(engl.) *Architecture Component*

Logischer Bestandteil des zu bearbeitenden *RFG*. Ziel der Arbeit mit OpenCAGE ist es, den RFG vollständig in überschneidungsfreie Architektur-Komponenten aufzuteilen.

Atomic Component (AC)

(engl.) *Atomic Component (AC)*

Eine Art von *Composite-Knoten*. AC's dienen der Strukturierung des dem *RFG* zugrundeliegenden Programms. Durch sie ergeben sich *Hierachieebenen* zwischen *Knoten*.

Attribut

(engl.) *Attribute*

Jeder *Knoten* und jede *Kante* besitzt eine Liste von *Attributen*. Je nach Typ sind verschiedene Attribute möglich.

Auswahlliste

(engl.) *Selection*

[GUI] Liste von Einträgen, aus denen einer ausgewählt werden kann (siehe Spezifikation Kapitel GUI-Konzepte, Selektieren).

Benutzer

(engl.) *User*

Anwender von OpenCAGE.

Bubble-Help

(engl.) *Bubble Help*

[GUI] Tooltip, mit kurzer Erläuterung. Dieser Hilfstext erscheint visuell hervorgehoben, wenn der *Pointer* vom Benutzer längere Zeit auf einer GUI-Komponente liegt, zu der eine Bubble-Help existiert.

Busy-Pointer

(engl.) *Busy Pointer*

[GUI] Mauszeiger während der Ausführung aller Aktionen, die länger als 1 Sekunde dauern, wenn der *Pointer* über das Programmfenster bewegt wird, in dem diese Aktion ausgeführt wird.

Checkbox

(engl.) *Checkbox*

[GUI] GUI-Element, mit dessen Hilfe der Benutzer den Status einer Option unabhängig vom Status anderer Optionen setzen oder rücksetzen kann.

Clipboard-View

(engl.) *Clipboard View*

Eine *View*, in der *RFG-Objekte* abgelegt werden, wenn sie von einer View in eine andere kopiert oder verschoben werden.

Composite-Knoten

(engl.) *Composite Node*

Knotenart. Im Gegensatz zu *Entitätsknoten* können *Composite-Knoten RFG-Objekte* enthalten.

Conflict-Highlighting

(engl.) *Conflict Highlighting*

Wenn es während einer *RFG-Manipulationsoperation* (z.B. *Accept*) zu Konflikten kommt, werden die *RFG-Objekte*, die den Konflikt auslösen, in allen Fenstern der *View*, in der sie vorkommen, in einem vom Benutzer definierbaren *Pixmap*/Farbhinterlegungen dargestellt. Der *Benutzer* muss diese Markierung wieder explizit entfernen.

Dokumente

(engl.) *Documents*

Wenn von Dokumenten gesprochen wird (Plural), dann sind damit folgende Produkte des Projekts gemeint: Spezifikation, Entwurf, Handbuch und der *Quelltext*.

Drag & Drop

(engl.) *Drag & Drop*

[GUI] Bezeichnet eine Kopier- oder Verschiebeaktion, die mausunterstützt durchgeführt wird. Als Metapher wird dabei das Aufnehmen und Wiederablegen benutzt.

Editor

(engl.) *Editor*

Der Editor ist ein externes ausführbares Programm, welches der Bearbeitung von Dateiinhalten dient. Der Editor ist vom Benutzer frei wählbar. Er ist programmspezifisch und wird durch die Konfigurationsdatei festgelegt.

Entitätsknoten

(engl.) *Entity Node*

Stellt im *RFG* vorhandene elementare Einheiten wie Typen, Variablen, Konstanten, Subprogramme usw. dar, die Elemente des zugrundeliegenden Systems sind.

Error-Log-Console

(engl.) *Error Log Console*

Spezielles Programmfenster, in dem alle Fehlermeldungen und Warnungen ausgegeben werden.

Fenstereintrag

(engl.) *Window Entry*

Ein Fenstereintrag im *Window-Master* referenziert ein Fenster, das eine *View* oder eine *Atomic Component (AC)* anzeigt.

Fensterparametersatz

(engl.) *Window Parameter Set*

Größe des Fensters, Position des Fensters, *Filter*, Knotenpostionen und *Zoom* eines bestimmten Fensters.

Filter

(engl.) *Filter*

Ein Filter legt aufgrund des Typs und/oder von *Attribut*en fest, ob bestimmte *Knoten* und *Kante*n dargestellt werden.

Filter-Window-Pointer

(engl.) *Filter Window Pointer*

[GUI] *Pointer* während der Filter-Modus aktiv ist.

Fictitious Bauhaus Reengineering

(engl.) *Fictitious Bauhaus Reengineering*

Der Auftraggeber dieses Projekts, die Abteilung Programmiersprachen und Übersetzerbau, IFI, Universität Stuttgart.

Graph-Area

(engl.) *Graph Area*

Die Fläche innerhalb eines *Graph-Window*s, in dem der Graph dargestellt wird.

Grapheneintrag

(engl.) *Graph Entry*

Ein Grapheneintrag im *Window-Master* referenziert eine *View* oder eine *Atomic Component (AC)* einer View. Grapheinträge sind immer Wurzeln eines Unterbaums, d.h es existiert mindestens ein *Fenstereintrag* als Blatt

Graph-Window

(engl.) *Graph Window*

Ein Graph-Window ist ein Fenster, das einen Teilgraph einer *View* anzeigt und fest dieser View zugeordnet ist.

Graph-Window, aktuelles

(engl.) *Graph Window, active*

Das Graph-Window das den Fokus hat. Auf ihm werden die Operationen des Benutzers angewendet.

Graph-Window, read-only

(engl.) *Graph Window, read-only*

Ein *Graph-Window*, in dem die *Use-Case*s, die eine *RFG-Manipulationsoperation* durchführen, nicht ausführbar sind.

Graph-Window-Button

(engl.) *Graph Window Button*

Button innerhalb eines *Graph-Window*, mit dessen Hilfe graph-window-spezifische Aktionen ausgeführt werden können.

GUI

(engl.) *GUI*

Graphische Benutzeroberfläche.

Hilfebrowser

(engl.) *Help Browser*

Externes Programm zur Anzeige von HTML-Dateien.

Hierachieebene

(engl.) *Hierarchical Level*

Hierachieebenen innerhalb von *RFG-Objekt*en entstehen durch die Nutzung von *Atomic Component (AC)*-Knoten innerhalb einer *View*. *Knoten*, die logisch gesehen innerhalb eines AC-Knotens enthalten sind, befinden sich eine Hierachieebene unter diesen AC-Knoten.

Kante

(engl.) *Edge*

Bestandteil des *RFG*, die logische Beziehungen zwischen zwei *Knoten* darstellt. Sie hat einen Typ und eine je nach Typ unterschiedliche Anzahl an *Attribut*en.

Kanten, induzierte

(engl.) *Edge, Induced*

Ist Menge aller *Kante*n, deren Quell- und Zielknoten zu der Menge der selektierten Knoten gehören.

Kanten, unvollständige

(engl.) *Edge, Incomplete*

Stellt dar, dass ein *Nachbarknoten* existiert, der nicht in der aktuellen View enthalten ist.

Kanten, verwaiste

(engl.) *Edge, Orphaned*

Verwaiste Kanten, sind *Kante*n, die keinen Quell- und/oder keinen Zielknoten besitzen.

Knoten

(engl.) *Node*

Hauptbestandteil des *RFG*. Es gibt *Entitätsknoten* und *Composite-Knoten*.

Konfigurationsdatei

(engl.) *Configuration File*

Eine Konfigurationsdatei enthält Einstellungen zu OpenCAGE. Es gibt programm- und projektspezifische Optionen, die in unterschiedlichen Dateien abgespeichert werden (siehe Spezifikation Kapitel Programm-, Projektspezifische Einstellungen und Konfigurationsdateien).

Konsistenzprüfung

(engl.) *Consistency Check*

Prüfung auf semantische Korrektheit der am *RFG* vorgenommenen Veränderungen. Ist zum Teil in der RFG-Bibliothek integriert, wird aber auch zum Teil von OpenCAGE übernommen.

Kunde

(engl.) *Customer*

Auftraggeber des Projekts. In diesem Fall ist dies *Fictitious Bauhaus Reengineering*.

Layout

(engl.) *Layout*

Ein Algorithmus zur Anordung von *Knoten*. Er kann auch auf Teilmengen von *RFG-Objekte*n einer *View* angewandt werden. Er bestimmt die Position jedes Knoten innerhalb des *Graph-Window*.

Layout-Selection-Pointer

(engl.) *Layout Selection Pointer*

[GUI] *Pointer* während der Layout-Selection-Modus aktiv ist.

Layout-Window-Pointer

(engl.) *Layout Window Pointer*

[GUI] *Pointer* während der Layout-Window-Modus aktiv ist.

Mark-Highlighting

(engl.) *Mark Highlighting*

Darstellung der markierten *RFG-Objekte* durch eine vom Benutzer definierbare Farbe oder Pixmap.

Markieren

(engl.) *Mark (to)*

Markierte *Knoten* werden in allen *Graph-Window*s durch *Mark-Highlighting* hervorgehoben.

Masterbar

(engl.) *Masterbar*

[GUI] Eine Iconleiste in der Workbench

Meldung

(engl.) *Warning*

Ein (Warn-)Hinweis von OpenCAGE für den *Benutzer*.

Menü

(engl.) *Menu*

[GUI] Auflistung der zur Verfügung stehenden Aktionen. Gesteuert werden sie durch Mausaktionen oder *Shortcuts*.

Menüeintrag

(engl.) *Menu Entry*

[GUI] Eintrag in einem *Menü*.

Menüzeile

(engl.) *Menubar*

[GUI] Eine Leiste mit *Menü*s in der *Workbench*.

Menüeintrag, highlighted

(engl.) *Highlighted Menu Entry*

Angewählter Menüeintrag. Er wird visuell hervorgehoben.

Metrics-Window-Pointer

(engl.) *Metrics Window Pointer*

[GUI] *Pointer* während der Metric-Window-Modus aktiv ist.

Metrik

(engl.) *Metric*

Eine Abbildung von bestimmten RFG-Objekteigenschaften auf eine Zahl.

Modus-Master

(engl.) *Mode Master*

Modus-Masters dienen dazu, Operationen auszuführen, die auf die Graph-Area wirken und für die der Benutzer in einen Dialog Informationen auswählen kann.

Modus

(engl.) *Mode*

Bestimmt die *Operation*en, die in einer *Graph-Area* getätigt werden können.

Modus, permanent

(engl.) *Mode, permanent*

Modus den man durch die Buttons in der *Werkzeugleiste* umschalten kann.

Bisher sind dies der *Standard-Modus* und der *Zoom-Modus*.

Modus, temporär

(engl.) *Mode, temporary*

Modus in den man durch einen *Modus-Master* gelangt. Der gewählte Modus ist solange aktiv, bis der Benutzer in ein OpenCAGE-Fenster klickt. Temporäre Modi überlagern den gerade aktiven Modus.

Move-Pointer

(engl.) *Move Pointer*

[GUI] Mauszeiger während einer *Drag & Drop*-Aktion

Multikante

(engl.) *Multi Edge*

Mehrfachrelation zwischen *Knoten*.

Nachbarknoten

(engl.) *Neighbour Node*

Knoten werden als Nachbarknoten eines anderen bezeichnet, wenn sie mit diesem durch eine *Kante* verbunden sind.

Notizbuch

(engl.) *Notebook*

[GUI] Art eines Dialogs, der einem Panel mit Reitern (Rider) im "Card-Layout" in der Programmiersprache Java entspricht.

No-Parking-Pointer

(engl.) *No Parking Pointer*

[GUI] Aussehen des *Pointer* der dem Benutzer anzeigt, das der aktuelle permanente *Modus* hier nicht angewendet werden kann.

No-Pointer

(engl.) *No Pointer*

[GUI] Aussehen des *Pointer*, der dem Benutzer anzeigt, das der aktuelle temporäre *Modus* hier nicht angewendet werden kann.

OpenCAGE

(engl.) *OpenCAGE*

Open Computer Aided Graphic Editor. Softwaresystem, welches innerhalb dieses Projekts entsteht.

Operation

(engl.) *Operation*

siehe *Aktion*

Parameter

(engl.) *Parameter*

Einstellungen für *Analysen*, *Filter* und *Layouts*.

Paste-Pointer

(engl.) *Paste Pointer*

[GUI] *Pointer* während der Ausführung der Paste-Funktion.

Pixmap

(engl.) *Pixmap*

Graphisches Element (Bild), zur Darstellung eines spezifischen Sachverhalts.

Pointer

(engl.) *Pointer*

[GUI] Mauszeiger. Je nach Modus ändert er sein Aussehen. (siehe auch: *Busy-Pointer*, *Filter-Window-Pointer*, *Layout-Selection-Pointer*, *Layout-Window-Pointer*, *Metrics-Window-Pointer*, *Move-Pointer*, *No-Pointer*, *Paste-Pointer*, *Select-Pointer*, *Standard-Pointer*, *Unfilter-Pointer*, *Zoom-Pointer*)

Programmparameter

(engl.) *Program parameter*

Programmeinstellungen, die global für alle Projekte gültig sind. Die Programmparameter sind benutzerspezifisch.

Programmspezifisch

(engl.) *Program - specific*

Einstellungen abhängig vom Benutzer.

Projektion, baumartige

(engl.) *Projection - tree*

Stellt die logische Hierarchie der *Composite-Knotens* einer *View* in einem Fenster dar.

Projektion, flache

(engl.) *Projection - flat*

Stellt *Entitätsknoten* und *Composite-Knoten* dar, jedoch keine *Knoten*, die in Composite-Knoten enthalten sind.

Projektparameter

(engl.) *Project parameter*

Die Menge aller Einstellungen, die die Darstellung der Fenster und der Fensterinhalte definieren. Legen *Style*, *Visualisierung*, Fensterpositionen fest, und enthalten den *Fensterparametersatz*.

projektspezifisch

(engl.) *project - specific*

Einstellungen abhängig vom gerade bearbeiteten RFG.

Protokolldatei

(engl.) *Protocol file*

Eine besondere Datei, die die ausgeführten Aktionen aufzeichnet. Sie kann zur Ausführung als *Skript* geladen werden.

Quelltext

(engl.) *Source code*

Alle elektronischen Daten, die für die Erzeugung eines Programms benötigt werden.

read-only

(engl.) *read-only*

Der Benutzer kann eine *View read-only* setzen. Es sind dann keine *RFG-Manipulationsoperation*en auf dieser *View* ausführbar.

RFG-Manipulationsoperation

(engl.) *RFG Manipulating Operation*

*Use-Case*s, die auf den *RFG* schreibend zugreifen. Dies sind alle Use-Cases der Sektion RFG-Manipulation.

RFG-Objekte

(engl.) *RFG-Objects*

Knoten, *Kante*n im *RFG*.

RFG

(engl.) *RFG*

Resource Flow Graph; er ist ein Graph, der die verschiedenen in einem Programmsystem vorkommenden Resourcen und die Beziehungen zwischen diesen darstellt.

RFG, gesperrt

(engl.) *RFG, locked*

Ein *RFG* ist gesperrt, solange Analysen auf den Daten des RFGs laufen. In dieser Zeit können keine Daten in den RFG geschrieben werden. Wenn die Analyse beendet oder abgebrochen ist, wird der RFG wieder entsperrt.

Reject

(engl.) *Reject*

Verschiebt die selektierten *RFG-Objekte* aus einer *View* in die *Trashcan-View*.

Select-All-Button

(engl.) *Select All Button*

GUI-Element innerhalb eines Graph-Window, mit welchem der Benutzer die Aktion Select All Objects in Graph-Window ausführen kann.

Select-Highlighting

(engl.) *Select Highlighting*

Hervorheben von selektierten (*Selektieren*) *RFG-Objekte*n und Einträgen im View-Master, Graph-Window und *Window-Master*.

Select-Pointer

(engl.) *Select Pointer*

[GUI] *Pointer* während der Select-Modus aktiv ist.

Selektieren

(engl.) *Select (to)*

Ein oder mehrere *Knoten* werden zur Bearbeitung aus einer *View* ausgewählt; dies geschieht entweder durch Anklicken mit der Maus oder mittels anderer Kriterien wie *Metrik*, *Attribut* oder Name.

Selektion

(engl.) *Select (to)*

Eine Menge von selektierten *Knoten*

Shortcuts

(engl.) *Shortcuts*

Mit einem Shortcut wird eine im *Menü* vorhandene *Aktion* über eine Tastenkombination aufgerufen.

Sichtbarkeit

(engl.) *Visibility*

Eine Einstellung, die für jedes Fenster einzeln getroffen wird, und die festlegt, ob alle Attribute entweder sichtbar sind oder nicht.

Single-User System

(engl.) *Single User System*

Es kann maximal eine Person gleichzeitig mit OpenCAGE arbeiten.

Skript

(engl.) *Script*

Elektronisches Protokoll der Interaktionen mit OpenCAGE, das zur Laufzeit aufgezeichnet wird, um es später nochmals abzuspielen. Es simuliert dann alle Benutzereingaben.

Slider

(engl.) *Slider*

[GUI] Metapher für einen Schieberegler. Es kann mit Hilfe des Sliders der Wert einer arithmetischen Größe eingestellt werden.

Source-Window

(engl.) *Source Window*

Programmfenster, in dem der zu einem *Entitätsknoten* gehörende *Quelltext* angezeigt wird.

Standard-Modus

(engl.) *Standard Mode*

Modus, in dem sich das System befindet, wenn kein anderer Modus aktiv ist.

Standard-Pointer

(engl.) *Standard Pointer*

[GUI] Mauszeiger im *Standard-Modus*, der normale Mauszeiger (Pfeil).

Statuszeile

(engl.) *Statusbar*

Der unterste Bereich in der *Workbench*. Dort werden bestimmte Zustandsinformationen dargestellt.

Style

(engl.) *Style*

Er bestimmt das Aussehen der *Knoten* und *Kante*n, die Lage und das Aussehen der *Attribut*e und die Art der Darstellung einer *Metrik*.

Symbolleiste

(engl.) *Iconbar*

[GUI] Enthält Icons, die Abkürzungen auf einen *Menüeintrag* darstellen.

System

(engl.) *System*

Das mit OpenCAGE zu analysierende Programm.

Trashcan-View

(engl.) *Trashcan View*

Enthält die *RFG-Objekte*, die aus der aktuellen *User-View* durch die Aktion Reject entfernt wurden. Sie kann mit einem speziellen Befehl geleert werden.

Unfilter-Pointer

(engl.) *Unfilter Pointer*

[GUI] *Pointer* während der Unfilter-Modus aktiv ist.

Use-Case

(engl.) *Use-Case*

In UML (Unified Modelling Language) definierte Form, um Programmabläufe als Sequenz von Interaktionen zwischen OpenCAGE und *Benutzer* zu spezifizieren.

Use-Case-Diagramm

(engl.) *Use-Case Diagram*

In UML (Unified Modelling Language) definierte Form, um bestimmte *Use-Case*s, ein Teilsystem und *Benutzer* dieser Use-Cases graphisch darzustellen.

User-View

(engl.) *User View*

Enthält die vom *Benutzer* als korrekt angesehenen *Architektur-Komponenten*n.

User-View, aktuelle

(engl.) *User View, active*

Ist die vom *Benutzer* festgelegte momentane *User-View*, in die alle vom Benutzer akzeptierten RFG-Objekte von OpenCAGE verschoben werden.

View

(engl.) *View*

Eine View ist eine Teilmenge von *RFG-Objekte*n eines *RFG*. Views sind Bestandteile des RFG.

View-Master

(engl.) *View Master*

Ein Programmfenster zur Verwaltung aller *View*s.

Viewmenge

(engl.) *Viewset*

Eine Menge von *View*s.

View, gesperrt

(engl.) *Locked*

Der *RFG* ist während *Analyse*n gesperrt. Der *Benutzer* kann nicht durch *RFG-Manipulationsoperation*en auf dem RFG zugreifen, solange er gesperrt ist. (siehe auch *Graph-Window, read-only*)

Visualisierung

(engl.) *Visualization*

Bestimmt, ob bestimmte *Attribut*e sichtbar sind oder nicht.

Werkzeugleiste

(engl.) *Toolbar*

[GUI] Enthält Icons, mit denen man den permanenten Modus wechseln kann.

Window-Master

(engl.) *Window Master*

[GUI] Ist ein Programmfenster, im dem eine Übersicht über alle geöffneten *View*s und *Graph-Window*s dargestellt wird.

Workbench

(engl.) *Workbench*

[GUI] Die Kommandozentrale. Hier können alle Einstellungen und Aktionen getätigt werden. Außer der *Menüzeile* gibt es noch eine *Symbolleiste* und eine *Statuszeile*. Die Workbench ist während des gesamten Programmablaufs geöffnet.

Zoom

(engl.) *Zoom*

Mit einem Zoom werden alle in der Graph-Area visualisierten *RFG-Objekte* um einen bestimmten Faktor vergrößert oder verkleinert.

Zoom-Pointer

(engl.) *Zoom Pointer*

[GUI] *Pointer* im *Zoom-Modus*.

Zoom-Modus

(engl.) *Zoom Mode*

Der Zoom-Modus kann durch Klicken auf das Zoom-Symbol in der *Werkzeugleiste* aktiviert werden. Er wird durch den *Zoom-Pointer* angezeigt.

Anhang C. Glossary

Accept

Operation which moves all selected RFG-Objects (including all RFG-Objects located in the hierarchically structure below the selected RFG-Objects) into the current User View.

Action

Interaction between user and OpenCAGE. It is specified by a Use-Case.

Affinity Threshold

A value configurable by the user, which is used to decide whether two nodes are identical or not.

Analysis

Function that tries to detect architectural information in the RFG. The gathered data will be stored in a new view.

Annotation

Optional text which describes an element of OpenCAGE (e.g. nodes, edges, ...)

Architecture Component

Logical integral part of the handled RFG. Main goal of the work with OpenCAGE is to divide the whole RFG into non-overlapping architecture components.

Atomic Component (AC)

Special type of Composite Node. It is used to give the analyzed program a proper structure. It builds up the hierarchical levels between the nodes.

Attribute

Nodes and edges have a wide variety of attributes. Different node and edge types have different numbers and types of attributes.

Bubble Help

[GUI] Tooltip with a short description. The help text is displayed as long as the pointer rests over a GUI-object. Bubble help offers help.

Busy Pointer

[GUI] Pointer that is shown while an action which takes longer than 1 sec is executed in the current window.

Checkbox

[GUI] GUI component which provides an any-of-many-choice.

Clipboard View

View where RFG-Objects are stored when the user has executed a copy or cut action.

Composite Node

Nodetype which may contain other RFG-Objects.

Configuration File

A File that contains all preferences for the user. There are different types of preferences: project specific ones and programm specific ones. They are stored in different files.

Conflict Highlighting

If a conflict/error occurs during an RFG manipulation operation (eg. Accept) all RFG-Objects that are involved in this error are highlighted, i.e. they are visualized as pixmaps specified by the user. The user has to remove the conflict highlighting of the affected nodes.

Consistency Check

Verification for semantic correctness that checks all changes done on the RFG.

Customer

Principal of the project, in this case Fictitious Bauhaus Reengineering.

Documents

In this case: specification, software design, user manual and sourcecode.

Drag & Drop

[GUI] Describes a copy or move operation done by using the mouse.

Edge

Integral part of the RFG. Logical connection between two nodes. Edgetypes may have different numbers of attributes.

Edge, Incomplete

Special type of edge, that points to a neighbor in another view.

Edge, Induced

All edges between a set of selected nodes.

Edge, Orphaned

This is a edge without a source and/or destination node.

Editor

External program which allows the user to edit a file. The location and name of the editor is specified by the user in the configuration file.

Entity Node

Represents basic RFG units eg. types, variables, constants, subprogramms of the system which is analyzed.

Error Log Console

Special window which displays errors and warnings raised by OpenCAGE.

Fictitious Bauhaus Reengineering

The customer of this project, the department of Programming Languages and Compilers, University of Stuttgart.

Filter

A set of criteria. OpenCAGE hides objects that do not match those criteria.

Filter Window Pointer

[GUI] Pointer that is shown while filter mode is active.

GUI

Graphical User Interface.

Graph Area

The area of a Graph Window used to display a graph.

Graph Window

Window that is used to display parts of a view and which is linked to that view.

Graph Window Button

Button in a Graph Window. It provides Graph Window specific operations.

Graph Window, active

Graph window where the operations/options from the user input are executed. It is the window that has the focus of the window manager.

Graph Window, read-only

Graph Window where Use-Cases that contain RFG-Manipulations cannot be executed.

Graph Entry

Special list item in the Window Master that points to a view or an atomic component of a view. A Graph Entry is always the root of a tree and has at least one leaf as *Window Entry*.

Help Browser

External program used to display HTML-Files.

Highlighted Menu Entry

Selected menu entry. It is highlighted.

Hierarchical Level

Hierarchical levels occur in the context of atomic components. Nodes that are logically within an atomic component share the same hierarchical level.

Iconbar

[GUI] Contains a bunch of icons, each representing a shortcut to a menu entry.

Layout

Algorithm to rearrange nodes. It can also be used on subsets of RFG-Objects within a view. It calculates the new positions of the nodes in the Graph Window. There is a set of layout algorithms shipped with OpenCAGE.

Layout Selection Pointer

[GUI] Pointer that is shown while the layout selection mode is active.

Layout Window Pointer

[GUI] Pointer that is shown while the layout window mode is active.

Locked

The RFG is locked while an analysis is executed. The user cannot perform RFG-manipulating actions on the RFG while it is locked.

Mark (to)

Marked objects are highlighted in any Graph Window showing these objects.

Mark Highlighting

Appearance of marked RFG-Objects with a user-defined color or pixmap.

Masterbar

[GUI] An iconbar in the workbench.

Menu

[GUI] List of operations. Handled by mouse or shortcut.

Menubar

[GUI] The bar in the workbench where menus are positioned to.

Menu Entry

[GUI] Entry in a menu.

Metric

Mapping between RFG-Object attributes and numbers.

Metrics Window Pointer

[GUI] Pointer that is shown while the metric window mode is active

Mode

Specifies the actions/operations which can be performed in the Graph Area.

Mode Master

The Mode Master is used to run operations selected by the user on a Graph Area. They are displayed in a dialog.

Mode, permanent

Mode that can be switched to by using the buttons in the toolbar. There are standard mode and zoom mode.

Mode, temporary

Mode that can be switched to by using a Mode Master. The chosen mode is active until the user clicks an OpenCAGE-Window. Temporary modes always replace the current active mode.

Move Pointer

[GUI] Pointer that is shown while a move operation is performed.

Multi Edge
: Multiple relation between two nodes.

Neighbour Node
: Neighbor Nodes are nodes that are connected by an edge.

No Parking Pointer
: [GUI] Pointer that indicates that the current permanent mode is not valid in this place.

No Pointer
: [GUI] Pointer that indicates that the current temporary mode is not valid in this place.

Node
: Integral part of the RFG. There are composite and entity nodes.

Notebook
: [GUI] Special dialog with tabs.

OpenCAGE
: Open Computer Aided Graph Editor. The Softwaresystem which will arise during this project.

Operation
: see action.

Parameter
: User adjustment for analysis, filter or layout.

Paste Pointer
: [GUI] Pointer that is shown while a paste operation is executed.

Pixmap
: Graphical element (eg. picture) used to display a special type of RFG-Object.

Pointer
: [GUI] Mousepointer. Its appearance depends on the active mode. There is a bunch of different pointers and also lot of pitchures of them.

Product Engineering
: The whole process to gather information about how and what to build for the customer.

Program - specific
: Settings that are set by user.

Program parameter
: Parameters that have a global validity and are userdefined.

project - specific
: Settings that are set for the current RFG.

Projection - flat
: Displays entitynodes and composite nodes, although not nodes which are contained in other composite nodes.

Projection - tree
: Displays the logical structure of composite nodes of a view in a graph window.

Project parameter
: A set of parameters that specify the appearance of windows and its content. They also specify the styles, visualization and window positions.

Protocol file
: A file which records all interaction with and produced by OpenCAGE. It can be used as a script afterwards.

RFG
: Resource Flow Graph. A graph that displays the different resources in program structures and its correlation.

RFG, locked
: see also locked

RFG Manipulating Operation
: Use-Cases that access the RFG by writing it.

RFG-Objects
: Nodes, edges in RFG.

read-only
: The user can switch a view to read-only mode. No RFG-Manipulation operations may be done in that view.

Reject
: Delete operation that moves the RFG-Object(s) from the current view to the Trashcan View.

Script
: A logfile that provides the information which is necessary to simulate the operations performed by the user. It is used for running a session without using the mouse.

Select (to)
: The user can select one or more RFG-Objects of the same view either by direct manipulation or by specifying criteria, e.g. attributes.

Select All Button
: Part of the Graph Window Buttons. This button allows the user to select all objects in this window.

Select Highlighting

Highlighting of the selected RFG-Objects and entries in the View-Master, Graph Window and Window-Master.

Select Pointer

[GUI] Pointer that is shown while the select mode is active.

Selection

[GUI] A set of selected objects, i.e. RFG-Objects, entries in the View-Master, entries in the Window-Master or list entries in dialogs.

Shortcuts

Maps a combination of keypresses to an action in a menu. A Hotkey.

Single User System

Only one person works with OpenCAGE at the same time.

Slider

[GUI] A slider is used to set an arithmetical value.

Source Window

A window that is used to display the source code belonging to an entity node.

Source code

Electronic data that is used to build a program.

Standard Mode

Mode which is active if none of the other modes are active.

Standard Pointer

[GUI] Pointer that is shown while the standard mode is active.

Statusbar

Area on the bottom of the workbench. It is used to display auxiliary comments/notes.

Style

Specifies the appearance of nodes, edges, the positions and look of the attributes and the appearance of displayed metrics.

System

The programm the user wants to analyze by using OpenCAGE.

Toolbar

[GUI] A window which contains the icons that are used to change the permanent mode.

Trashcan View

Contains RFG-Objects that were rejected by the user.

Unfilter Pointer

[GUI] Pointer that is shown while the unfilter mode is active.

Use-Case

Workflows that are defined in UML (Unified Modelling Language) as a sequence of events of an interaction between a user and OpenCAGE.

Use-Case Diagram

Graphical interpretation of Use-Cases in UML (Unified Modelling Language)

User

A person who uses OpenCAGE

User View

Contains all architectural components that were detected and verified by the user.

User View, active

The current User View defined by the user. OpenCAGE moves accepted RFG-Objects into this view.

View

A subset of the RFG-Objects of an RFG.

View Master

A window that controls all views

Viewset

A set of views.

Visibility

A setting that must be defined for each window. It defines whether an attribute is visible or not.

Visualization

Defines whether an attribute is visible or not.

Warning

A message from OpenCAGE

Window Master

[GUI] A window that provides an overview of all Graph Windows.

Window Parameter Set

Size, position of a window, applied filters, node positions and zoomfactor of a window.

Window Entry

A Window Entry in the Window-Master. Each Window Entry is linked to a Graph Window.

Workbench

Command center. A window which is opened during the complete workflow.

Zoom

An operation that changes the size of the graphical representation of the displayed RFG-Objects.

Zoom Mode

The Zoom Mode can be activated by pushing the zoom button in the toolbar of the workbench. Feedback is given by showing the zoom pointer.

Zoom Pointer

[GUI] Pointer that is shown while the zoom mode is active.