Per Sem Phone II (linked data framework)-001

Article · July 2014

CITATIONS
0

READS
35

1 author:

Georg Neubauer
Danube University Krems
40 PUBLICATIONS **0** CITATIONS

SEE PROFILE

PerSemPhone II

Kurzfassung:

Im Zuge des Projektentwurfs COIN-Projekt wurde ein Projektteil namens „*PerSemPhone II*“ abgegrenzt und wird hier näher beschrieben. *PerSemPhone II* stellt einen Searchengine mit integriertem Reporting dar. Als Teil des COIN-Projekt ist „*PerSemPhone II*“ der Kategorie Cyber-Intelligence zuzuordnen und dient vor allem der Suche nach Information im „*World Wide Web*“ und der teils automatisierten Aufbereitung der Daten durch die Erstellung von Reports. Die folgende Beschreibung dient der Analyse und der Elaboration der benötigten Iterationsschritte bei der Implementierung und ist sowohl für Interessenten als auch für Beteiligte bei der Erstellung von Wichtigkeit. Durch die Listung der Komponenten und dessen Interaktionskonzept sowie deren visualisierte Darstellung wird ein übergreifender Überblick bezüglich des Layouts (Designs) und der Funktionalität der Komponenten gegeben. Den Abschluss bildet eine Ausführliche Beschreibung der benötigten Methoden und Algorithmen zur Erstellung und Integration der Programmteile. Die Vorteile beim Anlegen des Projekts in Form der multivarianten Steuerbarkeit stellen im Fazit die Optimierbarkeit der Redundanz (Codierungs-Aufwand) gegenüber und gehen auf die Zukunftsorientierten Erweiterungkonzepte von „*PerSemPhone II*“ ein.

Georg Neubauer Dezember 2012

Inhaltsverzeichnis:

Beschreibung der Funktionsweise

Abgrenzung des Teilprojekts „*PerSemPhone II*“

Der Prototyp von „*PerSemPhone II*“ umfasst zunächst die Eingabemöglichkeit von URLs über eine Klasse Crawler-Thread „*cl_Crawler_v1_0*“. Jede Klasse enthält eine dieser URLs und kann durch die Definition als Thread-Klasse vom Programm parallel beliebige Inhalte von „URL-Adressen“ in Form von „plain/text“ speichern. Diese Informationen werden durch den Engine über eine Klasse Files aufbereitet „*cl_File_v1_0*“. Ein File ist ein Inhaltsabschnitt in einem Kontext der Datenaufbereitung. Der Kontext der gesammelten Inhalte wird über ein Knotenkonzept der Klasse Node realisiert „*cl_vNode_v1_0*“. Jeder Knoten kann mehrere Files beinhalten und ist an die Klasse Vektor gebunden „cl_Vector_v1_0“. Durch diese Art von Verknüpfung ist es möglich, ein Datenkonzept zu entwickeln, indem die Visualisierung der erstellten und gesuchten Inhalte mit speziellen Merkmalen der Gliederung einhergehen. Die Klasse Modul „*cl_Modul_v1_0*“ integriert diese Vektoren semantisch durch ein typologisches Signaturkonzept und visuell durch ein damit in Verbindung stehendes zweiphasiges Winkelsystem im dreidimensionalen Raum.

Die Funktionsweise der Module stellen einen weiteren Meilenstein der Projektabgrenzung dar. Funktional gesehen integriert ein Modul über ein zugrundeliegendes System der Klasse Graph „cl_Graph_v1_0“ das dessen Kern darstellt die entstehenden Inhalte der Ergebnisse des Crawlers.

Die Ergebnisse der Inhaltsanalysen aus dem Internet können in einer Klasse NodeContentView, der eine Teilansicht der Daten darstellt editiert, gelöscht, kopiert oder in Form eines Links in replizierter (der Computer stellt eine Referenz zum gespeicherten Ergebnis zur Verfügung) Form manipuliert werden und können im Dokumentenformat als Report exportiert werden.

Die NodeContentView stellt eine der drei Bearbeitungsebenen des Teilprogramms dar. Nach dem Bottom-Up-Prinzip stehen übergeordnet zwei weitere Ebenen der Visualisierung zur Verfügung, die eine Modulansicht *ModulView* und darüber eine metasprachliche Navigationsansicht *Meta*

Navigation bezüglich der Modulgruppen, die einem zeitorientierten DatenSpeicherKonzept der Klasse DataDisc „*cl_DataDisc_v1_0*“ unterordenbar ist.

Ein Datenselektionskonzept in Form eines dropable-Dialogs der Klasse Dropfield dient der barrierefreien Auswahl von Komponenten zwischen den drei ViewPorts und wird von einer Listenansicht bezüglich der Auswahl unterstützt.

Der Umfang der dem Teilprojekt „*PerSemPhone II*“ innerhalb des Coin-Projekts zuzuordnen ist, ist daher allgemein die Visualisierung und wie oben beschriebene Interaktion der drei Views sowie die Lauffähigkeit der Inhaltsanalysen bis zur Erstellung eines Reports.

Beschreibung des SearchEngines und dessen mögliche Konzepte

Die Suche mit „*PerSemPhone II*“ enthält drei Verschiedene Konzepte der Datenanalyse, die der sinnlichen Wahrnehmung angepasst sind. Die Anfrage der Suche als textuelles (charakterorientiert – Seiteninhalt), als grafisches (typorientiert Bildinformation) und als grafisches textuell reinterpretiertes Ergebnis in Form eines Screenshots des Inhalts einer Suchanfrage (WebSnipplet). Die Inhalte können über die Klasse Matrix „*cl_Matrix_v1_0*“ verglichen werden. Dazu dient ein Interaktionskonzept das auf dem Prinzip der Inhaltsgleichheit oder Inhaltsähnlichkeit basiert und über die Zusatzklasse Patrix „cl_Patrix_v1_0“ realisiert wird. Das Konzept kann so beschrieben werden, dass ein Inhalt über eine bitorientierte zweidimensionale Matrize auf inhaltliche Gleichheit geprüft werden kann. Dies geschieht über das vergleichen einer Patrize mit einer Matrize. Die Evaluierung gleichartiger Inhalte dient zum einen der Integration der Daten in Form der Differenzierung und zum anderen für die Reduzierung des Speicheraufwands in Form von Referenzen zu identen Inhalten. Die Differenzierung ist ein wichtiger Bestandteil des Integrationsprinzips bezüglich der Interaktion < Modul – Vektor >, auf welches ich später eingehen werde. Das Konzept der ternären Logik ermöglicht es, die Inhalte bei Ähnlichkeiten (prozentuelle Abweichung eines der verglichenen Inhalte voneinander) einem tieferen Unterscheidungskriterium zu unterziehen. Ist also das Ergebnis eines Vergleichs von Inhalten nicht ident <true, 1> oder nicht ident <false, 0> entspricht also z.B. die Reihenfolge der charakterorientierten Analyse zweier Inhalte nicht zu 0% oder zu 100% dem Inhalt, kann das Ergebnis <fuzzy, -1> d.h. 1% bis 99% übereinstimmen und weiter differenziert werden. (Um auf die Visualisierung einzugehen entspricht diese weitere Differenzierung einem Vektor-Insertment zwischen zwei bestehenden ähnlichen Inhalten). Anhand des ternären Konzepts, kann zu einem Großteil der Text aus Websnipplets extrahiert werden, indem vom Computer erkannt wird, dass Farbe und Anordnung eines pixelorientierten Screenshots bei der Analyse einen Hinweis auf textorientierten Inhalt enthalten.

Inhaltsaufbereitung durch „*PerSemPhone II*“ (Perspektivierung)

Die Ebenen der Datenverarbeitung sind grundsätzlich in drei Visualisierungs Screens gegliedert. Dazu ist anzubringen, dass das Visualisierungskonzept einem dreiteiligen Datentransformationsprinzip entstammt und die Visualisierungs Screens nur die hauptsächlichen Visualisierungs-Ebenen in Form von Viewports darstellen. Das eigentliche Datenkonzept ist das der Anordnung von Daten im dreidimensionalen Raum, indem jedem Punkt im R^3 auf der z-Achse DatenDiscs zugeordnet sind, die in der x-Achse in einer 1:m-Beziehung Module enthalten. Die zeitliche Komponente der DatenDiscs ist der y-Achse zugeschrieben. Innerhalb des R^3 kann jede der Komponeten in eine oder zwei der drei Richtungen gescrollt werden. (DatenDisc <y,z>; Modul <x>). Jedes Modul verfügt wie ein Folder über eine Bezeichnung bezüglich der Inhalte, der bei einem mouseover erscheint und über einen weiteren Dialog bezüglich der Verwendung verfügt, wählt man es aus. Die Module integrieren die Vektoren, deren Betrag wiederum eine zeitliche Komponente bezüglich der Erweiterung von bereits gesuchten und angelegten Inhalten ist. Damit ist nicht nur ersichtlich, wann Inhalte geändert wurden, sondern auch was der zuvor erstellte Inhalt war. Es ist daher möglich Rückblick auf entstandene Versionen zu halten und wurde von unserem Team mit der Bezeichnung „Content Evolution Tracking“ versehen. Die Inhalte selbst sind daher eine weitere Zoom-Stufe des Systems und sind anhand einer 1:m Beziehung durch Knoten „Nodes“ realisiert in die Einsicht genommen werden können. Durch die Vernetzungsmöglichkeit der beschriebenen Komponenten können ausgehend von DatenDiscs übergreifend gleichartige Komponenten entweder durch Auswahl in eine Liste als neues bearbeitbares Modul exportiert oder durch Verknüpfung von Knoten über die Klasse Links „*cl_NodeLinker_v1_0*“ in der NodeContentView verarbeitet, editiert und als Dokument in Form eines Reports ausgegeben werden.

Sentiments, Statistiken und Monitoring

Die Häufigkeit von gleichartigen Inhalten bildet den Ausgangpunkt bezüglich der Knotenanalyse und kann als Statistik ausgegeben werden. Über die tiefere Analyse anhand der ternären Logik können dann Sentiments und Inhaltsabweichungen bzw. Meinungen (Sentiments) über ein gesuchtes Thema erzeugt werden. Dies geschieht über den Matrizenvergleich über Inhaltsähnlichkeit durch integrierte Vektoren. Zudem ist es möglich die Veränderung bestimmter Inhalte einem Verfasser zuzuordnen (Monitoring) und die Erweiterung der Inhalte zu erfassen. Tendenzen zu einem Thema werden dadurch leicht erkennbar.

Eine Lochmatrize die modulorientiert ist, kann durch das Auftreten weniger Suchergebnisse Aufschluss über das „nicht Vorkommen“ eines Themas geben und beispielsweise so auf Innovationslücken weisen, die vom Nutzer kreativ analysiert werden können. (Content Evolution Tracking)

Durch die Inhaltsanalyse in Form des Vergleichs können Dokumentanalysen getätigt werden, die Hinweise über die Güte eines Werks ermöglichen oder auf den ursächlichen Verfasser verweisen. (User Specification)

Erstellung der Grundstruktur

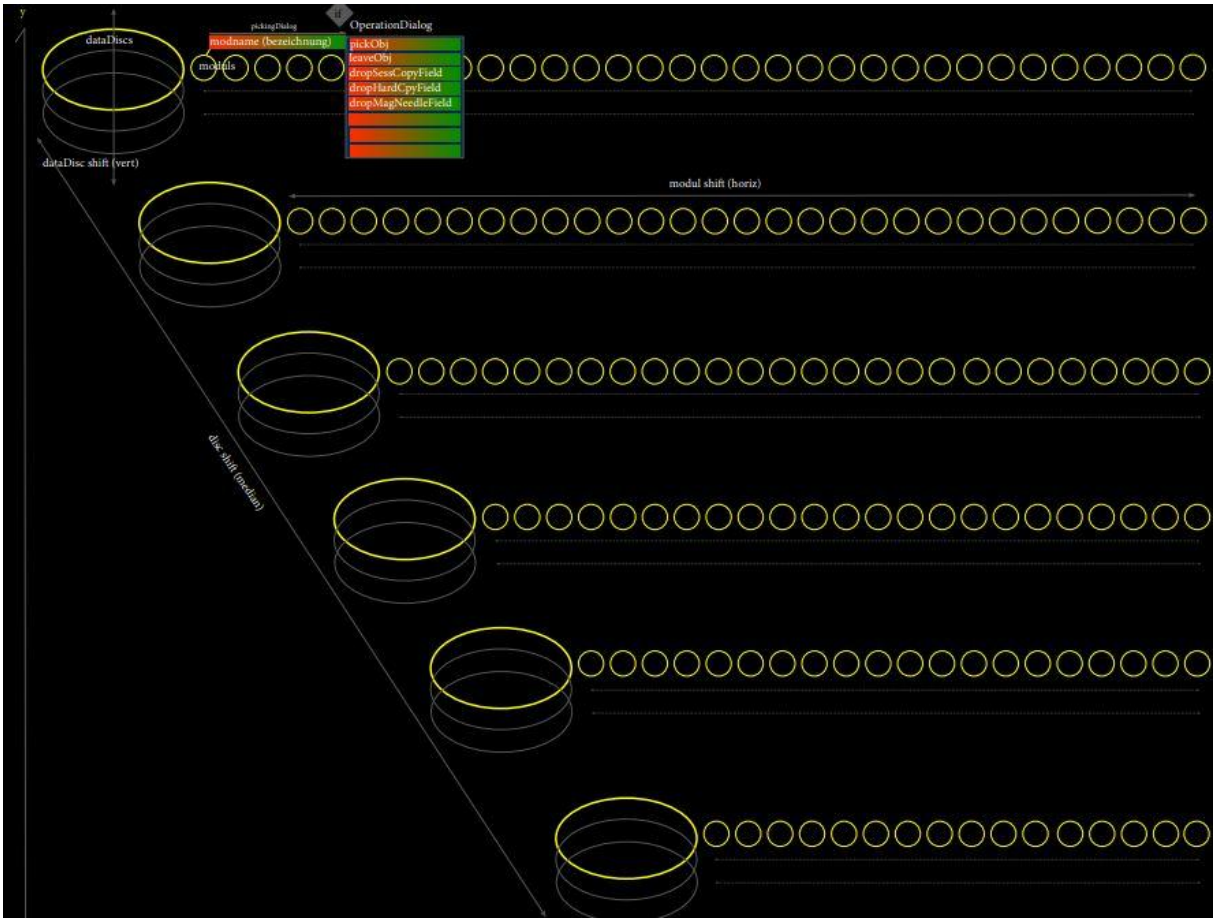
Das duale Komponeten –Paradigma

Der dualistische Ansatz beinhaltet eine Trennung der Typen in zwei Komponenten zu denen eines einen operationalen Ansatz als Komponente (interaktive Komponente) in Anwendung auf sein Gegenüber, einen statischen Ansatz als Komponente (statische Komponente) auf der interagiert wird, darstellt. Die Gemeinsamkeit des Objektentwurfs bezüglich der Komponenten wird durch deren Klassenentwurf erläutert. Eine Komponente innerhalb des Systems besitzt neben den spezifizierenden Eigenschaften objektorientierter Programmierung (Konstruktor, Methoden …) Aussagen über deren visuelle Beschaffenheit. Dazu wird jeder erstellten SystemKomponente innerhalb der Definition der Klasse zu der Typinformation (siehe Typologie des Signaturprinzips) deren Form und Farbinformationen als Eigenschaft eingeschrieben. Das erhöht die Übersichtlichkeit des Codes, denn die Anordenbarkeit innerhalb des Systems ist durch die generalisierte Form der Komponenten nicht beeinflusst. Durch Transformationen wie Skalierung und Rotation kann die notwendige Erscheinungsform innerhalb der Visualisierung leicht angepasst werden. Kann nun eine interaktive Komponente auf eine statische Komponente einwirken, so sind diese beiden Komponenten zumindest mit einer Methoden-Schnittstelle verknüpft. Die Differenz zwischen interaktiven und statischen Komponenten kann als diese Zuordnungsvorschrift gesehen werden. Der Vorgang der Erstellung einer Komponente wird über die Klasse Edges „cl_EdgesObj_v1_0“ und die Klasse Shape „cl_ObjShape_v1_0“ gemacht, indem diese Eigenschaften innerhalb der neuen Komponenten-Klasse instanziert werden. Damit ist eine beliebige Beschaffenheit möglich, weil über die callback-Struktur jedes grafischen 3D-Engines Polygone erstellt werden können. Der Vorteil eines individuellen Designs ist dadurch vorab gegeben.

Die Komponentenlogik anhand der Komponentenfunktionalität

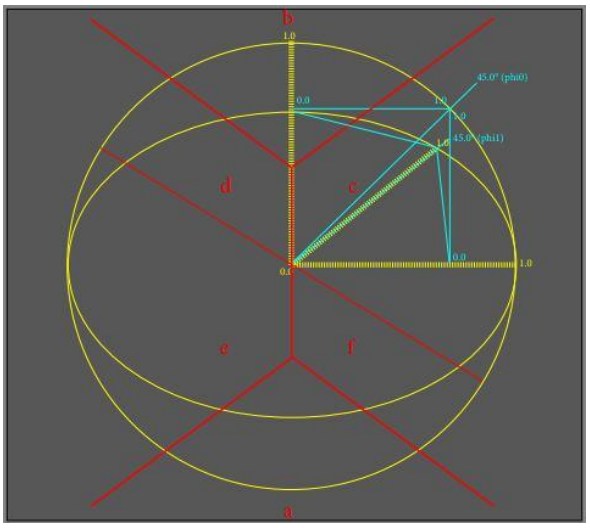
Statische Komponenten

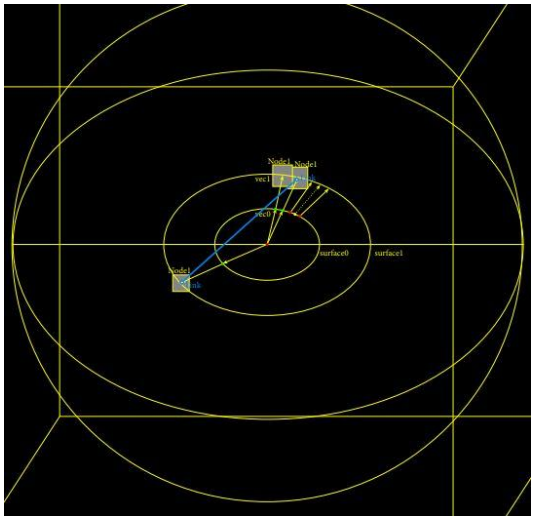
DataDiscs:



Die DatenDiscs stellen zeitlich gegliederte Elementstapel auf der z-Achse dar. Sie entsprechen einer Datenstruktur (z.B.: Serverknoten, Datenbank, Filesystem). Die Navigation erfolgt über die zeitorientierte Verschiebung auf der y-Achse oder die artorientierte Verschiebung auf der z-Achse. Die zeitorientierte Verschiebung begreift die zugehörigen Module mit ein und ist dadurch auch inhaltsorientiert. DataDiscs können selektiert werden und interagieren über drag und drop mit den Dropfields der Meta Navigations View.

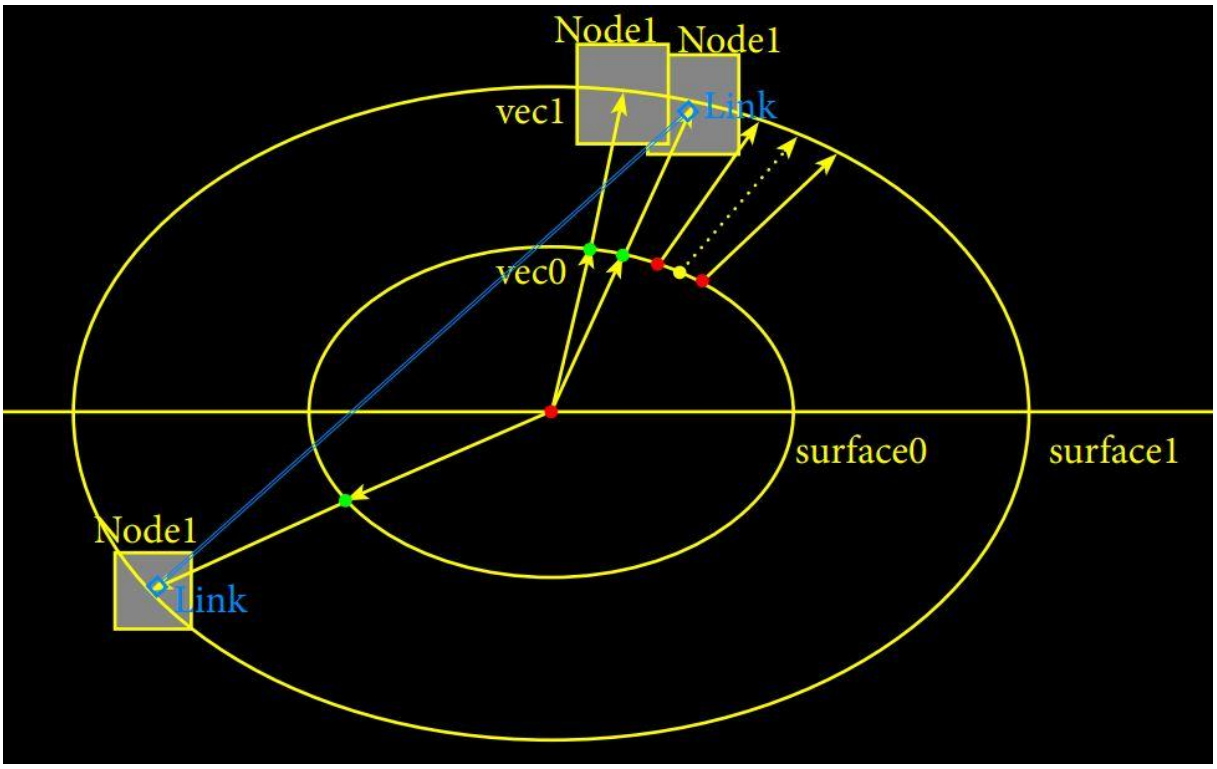
Module:





Die Module sind eine 1:m Zuordnung an jeweils eine DatenDisc und sind auf der x-Achse der jeweiligen zeitorientierten Gliederung der DataDiscs auf der y-Achse mitgeführt. Verschiebt man daher eine Version der DataDiscs auf der y-Achse, werden auch die jeweiligen Modulketten hervorgehoben und bilden die oberste Modulkette. Module können durch ein Maskenwerkzeug selektiert werden und interagieren über drag und drop mit dem Dropfield. Durch die Möglichkeit der Auswahl von Modulen aus verschiedenen DataDiscs, kann man sich anhand einer Liste der gewählten Module (Listenansicht) Überblick verschaffen.

Vektor:

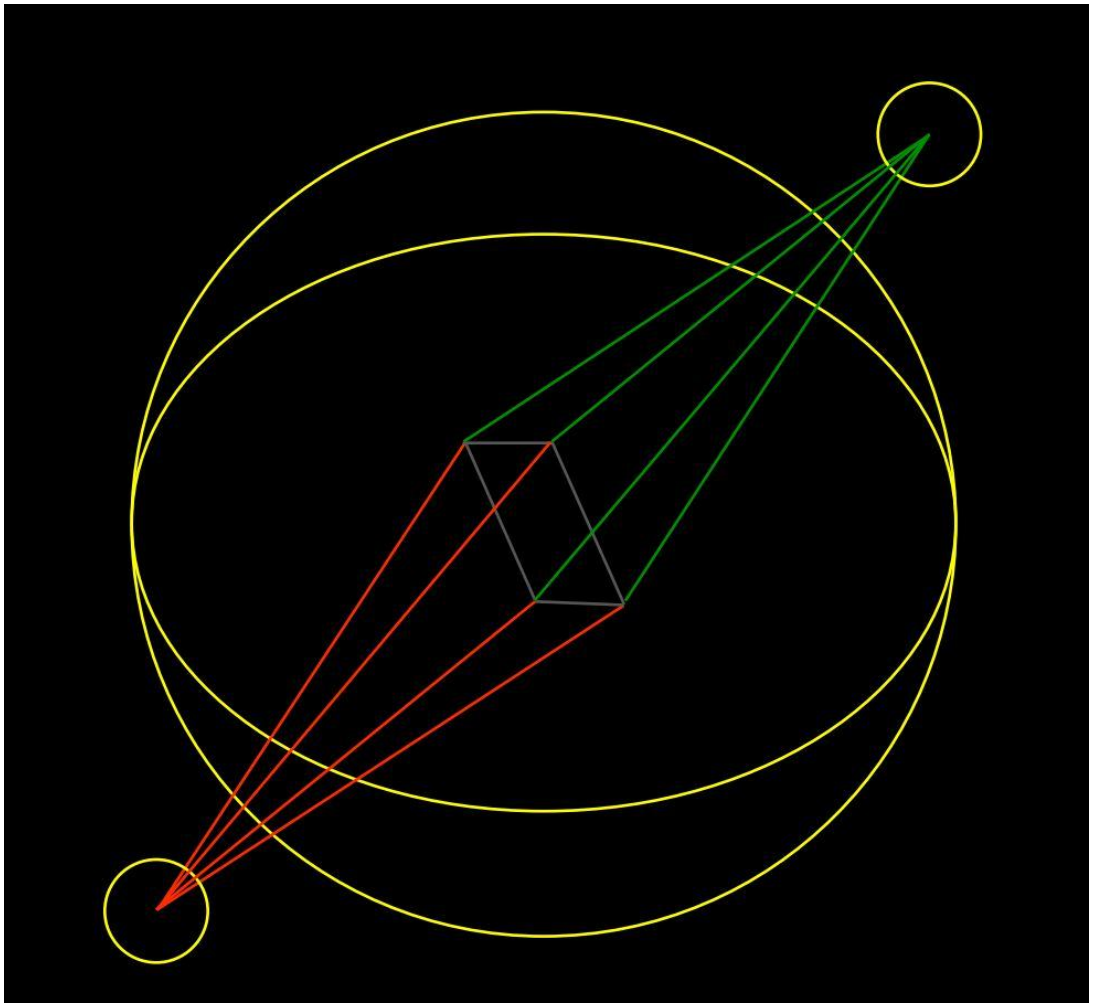


In der Modulansicht stellt der Vektor in der Modul View ein semantisches Ordnungskonzept der Inhalte dar, indem er in Betrag und Richtung nach dem oben erwähnten Konzept gleichartige Inhalte und deren Erweiterungen kennzeichnet. Der Schaft eines Vektors ist mit der ternären Logik versehen, damit eventuell eine Erweiterung eines Inhalts unter starker Änderung in einer zeitlichen Modulschicht neu angesetzt werden kann. Damit kann sein, dass der Gesamtvektor, der durch die zeitlichen Ergebnisse entsteht nicht im Ursprung des Moduls beginnt sondern dort wo im einen Fall das Ergebnis erstmals festgestellt wurde oder es im anderen Fall manuell erzeugt wurde oder der Inhalt zwischenzeitlich gelöscht wurde. Die Vektorspitze weist jeweils auf einen Inhalts Knoten des Systems. Damit ist auch jeder Teilvektor eines Gesamtvektors gemeint, der ja eine repräsentative frühere Version des Inhalts beinhaltet.

ModulSchicht:

Begrenzt eine Suchreihe in einem Modul zeitlich. Die Beträge der Vektoren sind in der Visualisierung gleicher Länge.

Transformations Eininheit:

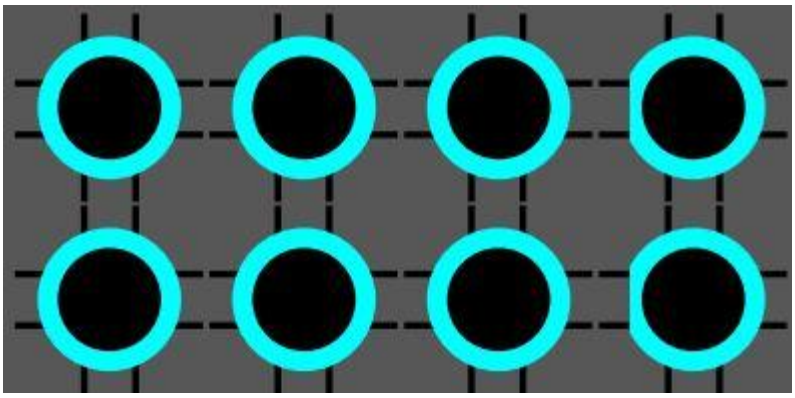


Über die Transformationseinheit kann ein jeweiliges Modul skaliert und rotiert werden. In der Darstellung wird dann das visualisierte Modul mit den angezeigten Vektoren gleichermaßen an die Befehle die die Transformationseinheit erhält mittransformiert. Der Vorteil ist dadurch, dass bei einer hohen Integrationsdichte von Vektoren diese durch die Transformationen leichter daraus zu selektieren sind und das Anklicken der Knoten ermöglichen.

Nodes (Knoten):

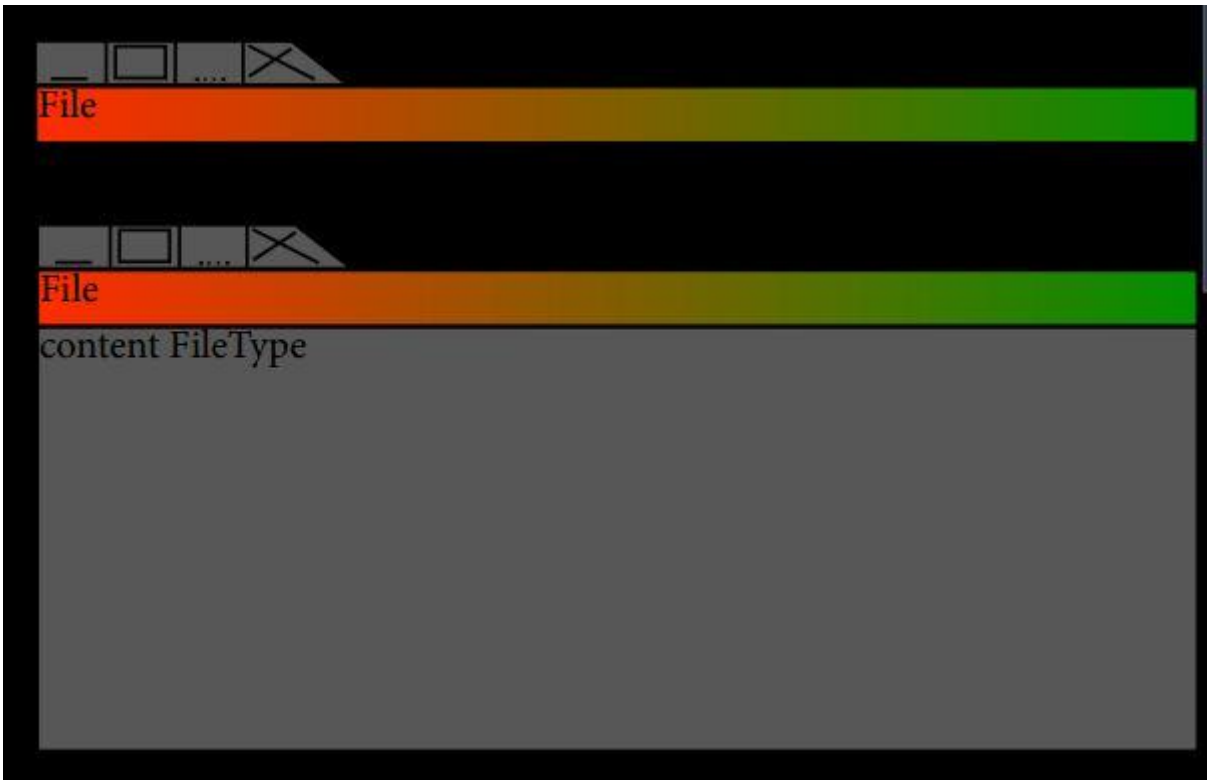
Die Knoten befinden sich an der Spitze der Vektoren und stellen innerhalb der Modul View die Abstraktion zu den Inhalten dar, die ausgewählt wurden. Während sie in der Node Content View die aus der Liste zusammengeführten Files in minimalisierter Form darstellen. Es gibt durch das Session-, HardCopy-Konzept zwei Views der zusammengeführten Listen, deren Herkunft bezüglich der sie beinhaltenden Module und übergeordnet DataDiscs erweiternd visualisiert werden.

Inhaltsobjekte:



Inhaltsobjekte sind eine Art Zwischenablage, für SignaturReferenzen (auf die Signatur selbst wird im nächsten Abschnitt eingegangen) sie enthalten wie die Module und Datadiscs eine Hervorhebung ihres Bezeichners bei einem mouseover. Sie sind ein Pool von Vorgefertigten, immer wieder benötigten Objekten und können zum editieren und erweitern der Files verwendet werden, indem man sie einfach auf die gewünschte Cursorposition im File setzt. Jeglicher typisierter Inhalt (Text, Grafik, Script, Statistik …) kann dadurch mit dem Inhalt des Files verlinkt werden.

File:



Files sind von der Netzstruktur des Systems abgeleitete Inhaltsklassen (Container). Sie verfügen über drei Zugriffsrechte, die wiederum über die ternäre Logik realisiert sind. Ein File kann beispielsweise im Falle eines Scripts neben den Modi 1 für Zugriffsrecht und 0 für kein Zugriffsrecht auch in einem Warnungsmodus existieren, wenn es ein nicht fertiges Script darstellt. Filelisten sind also Zusammenstellungen von Files aus Knoten (Nodes), die über ein entweder bereits ein auf dem Systemspeicher abgelegtes erzeugtes Modul darstellen oder die Filelist ist eine Zusammenstellung, die als Session angelegt wurde. Die Haupteigenschaften der Files sind ihre Editierbarkeit, Minimalisierung (als Ausgangsdarstellung im System gewählt), Maximalisierung, Löschung, Manipulierbarkeit, einzelne Exportierbarkeit und Exportierbarkeit als Fileliste in Form eines Reports.

Die List Komponenten:

Es gibt zwei verschiedene Listentypen bezüglich der Meta Navigations View und der Modul View. In der Meta Navigation erhält man sowohl eine Übersicht auf die Ausgewählten Module der HardCopy als auch auf die in der Session ausgewählen Module in Form von zwei verkleinerten Listen, die visuell von der DatenDisc auf das Modul verweisen und deren Bezeichner das jeweilige Modul benennen. Man kann sie nun auch wieder per drag und drop aus der List nehmen.

In der Modul View kann man zur beschriebenen Übersicht jedes Modul der Liste Auswählen, um eine bestimmte Auswahl von Knoten (Nodes) zu bekommen.

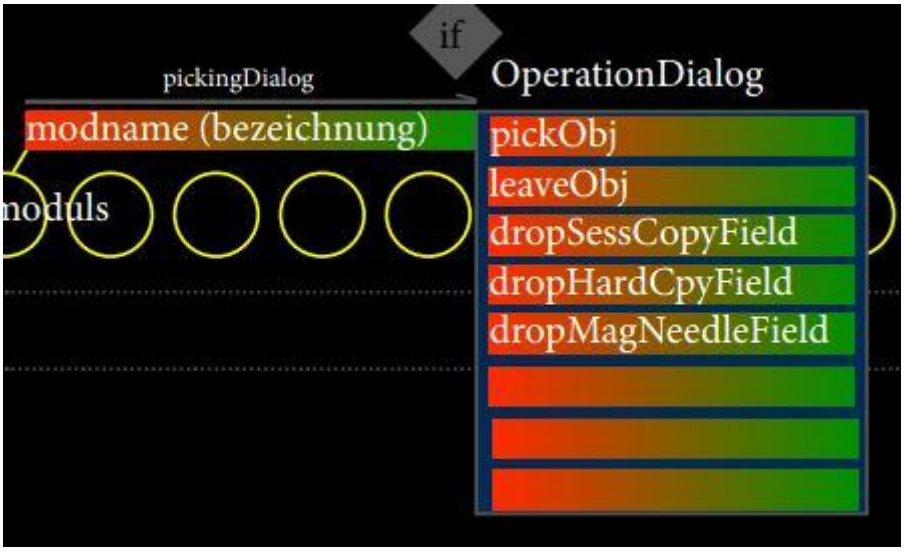
Die Listen beider Views enthalten die gleichen Module. Wird in der Modul View eine Auswahl bezüglich eines Moduls durch das selektieren mehrerer Knoten benötigt, wird das Modul in der Liste automatisch auf den Umfang der selektierten Daten angepasst.

Interaktive Komponenten

Dropfield:

Dropfields sind neben der menüorientierten Auswahl von Modulkomponenten vor allem dazu geeignet schnelle Datenselektionen ausführbar zu machen. Zieht man eine Komponente in das jeweilige dafür vorgesehene Dropfield, wird diese Komponente mit der nächsten Verarbeitungsebene verknüpft. Das Dropfieldkonzept basiert auf der Orientierung des Systems in zwei Grundzuständen. Zum einen gibt es Daten und Inhalte, die womöglich noch gebraucht werden oder deren Bearbeitung in einem unfertigen Zustand ist oder deren Orginalzustand nicht verändert wird. Deshalb wird sicherheitshalber zwischen den Zuständen Session und Hardcopy unterschieden. Für den jeweiligen Bedarf sind dann beide Dropfields innerhalb eines Viewports auswählbar.

Operation Dialog:



Wird der Bezeichner einer mit einem Operation Dialog ausgestatteten Komponente ausgewählt, klappt dieser auf, und bietet ein Untermenü mit Operationen an, die auf die Komponente angewendet werden können. Komponenten mit diesem Dialogmenü sind vor allem DataDiscs, Module, Nodes und Files und Inhaltsobjekte zum editieren.

TimeDate:

Jede der Komponenten enthält eine Zeitangabe bezüglich deren ersten Auftretens innerhalb des Systems. Diese Angabe wird bei Veränderung der gleichen Komponente nicht angepasst. „TimeDate Einträge können aber vor allem auch selbst angelegt werden.“

Typologie der Komponenten

Die Signatur der Komponenten von „*PerSemPhone II*“ stellt zuzüglich der TimeDate-Komponente ein vierteiliges Konzept dar. Es besteht aus einer Typkennung einer Identifikationszahl einem semantischen Bezeichner sowie einem je nach Komponente angewandten funktionalen Eintrag.

Der funktionale Eintrag dient in der einen Form der Anordenbarkeit von Komponenten innerhalb der Visualisierung als Positionskennung, als Versionskennung oder als über die Identifikationszahl indexierte Inhaltskennung.

Die Typkennung besteht aus alphanumerischen Zeichen, die bei Übereinstimmung mit dem Interpreter die Komponente einen Typ zuordenen und zur Interpretation der Komponente ein reduziertes, zur Darstellung des syntaktischen Inhalts der Komponente, Befehls-Set zur Verfügung stellen.

Die Identifikationsnummer ist ein fortlaufender Index, der die Menge der Dateien gleichen Typs innerhalb der Ordnung der übergeordneten Komponente beinhaltet.

Der semantische Bezeichner dient als eigentlicher Erkennungs-Tag für den Nutzer und sollte nach Möglichkeit eine eindeutige Beschreibung enthalten.

Die Ansteuerung der Komponenten sowie deren Gliederung (beispielsweise bei Suchergebnissen aus dem Internet) geschieht ausschließlich über die Typologie dieser Signatur.

ModulLink, NodeLink:

Über Links, können Verknüpfungen manuell festgelegt werden. Auf Basis der Nodes dient dies einer Vernetzung innerhalb eines Moduls. Dadurch kann ein Netzwerk von verlinkten Nodes in ein Dropfield gezogen werden. Man verlinkt die einzelnen Nodes in der Node Content View dabei vorab. Dies spielt bei der Automatisierung der Erstellung von Reports eine große Rolle.

In der Node Content View können nach dem gleichen Ergebnis die Files verschoben und integriert werden. Session Files können zusätzlich in die Hard Copy Files aufgenommen werden sowie das Verlinken und das Einfügen von Inhaltsobjekten auf bestimmten Positionen ermöglicht ist.

Syntaktische Beschreibung der typologischen Signatur:

a) Typenkennung:

<typ, code, txt, dia, url, css, …>

b) Identifikationsnummer:

<0 – inf.>

c) Bezeichner:

<[a-z], [0-9], <space, ., :, …>

d) funktionale Einträge

Positionierung:

Anhand eines Winkelsystems dass den Quadranten und die Raumrichtung des Vektors angibt

<a,b: front, back; c-f : Q1, Q2, Q3, Q4; phi0: [0-inf.], phi1:[0-inf] 0>

(Die erhaltenen Winkelzahlen und Quadranten hängen mit den Ergebnissen der Matrixvergleiche bezüglich der Inhalte zusammen)

Versionierung:

<v1_0 – vinf._inf.> Standardbelegung einer erstellten Komponente.

Für einzelne DataDiscs, Module und Files gedacht.

Indexierte Inhaltskennung:

<typ-idNr1 – typ-idNr2> Die Komponente beinhaltet eine Menge von Daten des gleichen Typs dessen Index von der Identifikationszahl1 bis zur Identifikationszahl2 reicht.

Implementierung und Erweiterbarkeit der Typen

„*PerSemPhone II*“ ist so konzipiert, das Komponenten neu erstellt und hinzugefügt werden können. Dazu muss eine Typkennung angelegt werden, die von den anderen bereits existierenden Komponenten abweicht. In der Visualisierung ist dies nur innerhalb des beschriebenen Rahmens als Erzeugung eines Inhaltsobjekts möglich, da der Interpreter wissen muss, wie die Komponente zu interpretieren ist.

Fazit

Optimierung vs. Usability (Vorteile des Meta-GUI)

Das Meta-GUI hat besondere Vorteile in der Geschwindigkeit, in der Daten selektiert und bearbeitet werden können. Das offene Konzept von „*PerSemPhone II*“ ist vor allem deshalb so wichtig, weil die Steuerbarkeit der Komponenten und die Verwaltung der Daten eine Lösung auf Basis einer Gestensteuerung oder einer durch Augenfocus gesteuerten Alternative einbegreifen. Dies unterstreicht die Innovation des Konzepts. Die Komponenten sind der multivarianten Benutzungsart angepasst. Es gibt mehrere Wege, Datensets über das dreiteilige GUI zur Verarbeitung zu holen (Operation Dialog, Dropfields, Konsoleneingabe). Durch das Verlinken können schnell Netzwerke von Modulen oder Knoten erstellt werden, die zur weiteren Bearbeitung zur Verfügung stehen.

Verwendungsmöglichkeiten des Engines (Planung – Controlling)

Planung

Das Prinzip der Planbarkeit ist nicht nur für strategisch angesetzte Suchvorgänge dienlich. „PerSemPhone II“ dient vor allem der Planung von Terminen über die Verwendung von Modulen. Um das Konzept zu illustrieren werden dabei drei Zeitstufen verwendet. (Past, now, future). Unter der Vergangenheit sind bereits erhaltene Suchergebnisse in den DataDiscs gespeichert und in den jeweiligen Modulen enthalten. Momentane Vorgänge stellen die „Top View“ der Meta Navigation View dar, zu der durch Anklicken des Symbols immer wieder zurückgekehrt werden kann. Die Planung eines Ereignisses gestaltet sich dann folgendermaßen. Die Terminplanung kann über eigens angelegte TimeDate Module eingetragen werden und befinden sich über dem momentanen Zeithorizont (Top View). Die Differenz zwischen der Momentanzeit und dem anglegten Termin, wird dadurch immer geringer (Anordnung DataDisc mit Termin Modulen bezüglich der y-Achse). Der Betreff und eventuell vorab wichtige Informationen können in die Nodes eingetragen werden.

Controlling

In der heutigen Zeit ist das Controlling der wichtigste Job bezüglich der Wirtschaft. Der Controller muss defacto nicht nur wissen wann etwas getan werden soll, sondern er muss einen Überblick haben, was von Operatern getan werden soll und dies sollte möglichst genau das sein, was der Operator tut. Durch die Freigabe von Daten über die Zugriffsrechte erhält der Controller diese Möglichkeit. Der Operator kann dadurch wissen, welche Arbeitspakete er abzuarbeiten hat. Er hat die Möglichkeit sich einen Überblick über das Gesamtprojekt zu verschaffen, kann jedoch nur die für ihn vorgesehenen Arbeitsabschnitte (Files) bearbeiten, die ihm vom Controller zur Verfügung gestellt wurden.

Wenn es möglich ist, sollte der Operator vor allem mehrere unterschiedliche Arbeitspakete erhalten, um innovativ und übergreifend arbeiten zu können. Dies ist durch „*PerSemPhone II*“ ermöglicht.

Ansatz einer Konsolenorientieren Scriptsprache

Die Integration einer Konsole ist aus mehreren Gründen sehr wichtig. Vor allem jedoch deshalb, weil die Signaturen des Systems sich bestens dazu eignen syntakische Arraycopy-Routinen bezüglich der Komponenten zu verwirklichen. Wenn man also den Knoten 0 bis 127 verlinken will, gestaltet sich dies in der Konsole als nur ein Befehl. (pseudocode: link nod_000 – nod_127 je nachdem welche der vier Signaturenteile herangezogen werden). Durch die eindeutige Kapselung gleicher Typen ist dies nun möglich. Das Anlegen von neuen Typen und die Typerweiterung sollte für die Operator durch die Konsole ermöglicht werden.

Lösungsansatz der Varianz von Daten-Stream-Konzepten

Um den Umfang der Möglichkeiten des Systems etwas zu elaborieren wird hier darauf eingegangen, dass das Daten-Stream Konzept über das Erstellen von Screenshots (Websnipplets) auch auf framebasierenden Medien wie Video oder Audio angewendet werden kann. Dazu ist eine Typerweiterung der Module angedacht worden, die Audio- und Videoformate bei der Analyse in angepasste einzelne Einheiten zerlegt und deren Inhalte in Form eines zeitlich orientierten Modulstapels visualisiert und editieren lässt.

Georg Neubauer Dezember 2012