eewcoSim

Dokumentation

Version:
v_experimental 2012_01 [7][3]

Version der Dokumentation [7]
Version eewcoSimchart [3]

Lizenz

Inhaltsverzeichnis

Zielgruppe der Dokumentation

eewco-Modellierer

Die Dokumentation wendet sich in erster Linie an Forscher, die eewco-Modelle programmieren möchten. Dazu wird die Programmstruktur erläutert. Die Programmstruktur kann als Ausgangspunkt für die Diskussion und Weiterentwicklung der Programmierungsmethodik oder für eigene Projekte genutzt werden.

Ziel ist zum einen eine belastbare Programmierungs- und Simulationsumgebung zu bekommen. Es sollte zudem gewährleistet sein, dass Programme anderen Forschern zur Verfügung stehen.

Modelle befragen

Die Dokumentation sollte es auch ermöglichen, ein programmiertes Modell, das in Form eines Datenbankdumps vorliegt, befragen zu können. Dazu sind die Kenntnisse zum Einrichten einer Datenbank („Die ersten 10 SQL-Statements“) notwendig.

Wenn eigene Variablen mit der eewcoSimchart-Zeitreihendarstellung abgefragt werden sollen, dann ist eine MySQL-Stored-Procedure nach Vorbild zu schreiben. Dazu sind die Abfrage-SQL-Statements notwendig, und soviel Verständnis von der Programmstruktur der Vorlagen-Procedure, dass die Funktionen erhalten bleiben.

Diese Dokumentation ist keine Einführung in die Programmierung von SQL und Python. Hinweise, wie entsprechende Einführung zu gestalten wären oder Hinweise aus entsprechende Informationsmaterialien werden gerne aufgenommen.

Allgemeine Anforderungen

Anforderungen, die aus eewco-Modellen folgen

- Simulation diskreter Zeitperioden
- vielfältige Weiterentwicklungen und Überarbeitungen ermöglichen

Anforderungen, die sich aus dem Wunsch nach Verbreitung und Entwicklung ergeben

- Kostengünstiger und zeitgünstiger Zugang zur Software
- Zumindest in der Grundform an die verbreiteten Rechnerkapazitäten anpassen oder für Zugang zur benötigten Kapazität sorgen
- Beachte dazu auch Zeitaufwand für Simulationsläufe zum Debuggen, Szenarioanalyse und zur Diskussion
- Die Software sollte zuverlässig verfügbar sein.

- Kommunizierbarkeit: Das Programm sollte auch Nicht-Programmierern soweit erläutert werden können, dass sie die Arbeitsweise verstehen.
- Die Programme sind so zu gestalten, dass sie von anderen für ihre Fragestellungen weiterverwendet oder angepasst werden können.

Gewählte Softwarestruktur

Simulationsprogramm und Datenspeicherung: MySQL-Server

Das Simulationsprogramm wird in dem Datenbankserver *MySQL* realisiert, in der Sprache MySQL-*Stored Procedures*.

Die Datenbank als Softwarebasis bietet die folgenden Vorteile:

Die Werte eines Simulationslaufs werden in Tabellen gespeichert und können später auch im Hinblick auf Fragestellungen befragt werden, die beim Durchführen des Simulationslaufs noch nicht interessiert haben.

Die Speicherung der Simulationsdaten in den Tabellen erleichtert das Nachvollziehen des Programmablaufs. Dies ist zum einen für das Debuggen wertvoll. Als besonderer Vorzug könnte sich auch erweisen, dass das sukzessive Ausfüllen von Tabellen auch von Personen nachvollzogen werden kann, die nicht mit der Programmierung vertraut sind.

Programm und Daten werden zusammen abgespeichert.

Datenabfrage und Programmierung: die Workbench zu MySQL

Die *Workbench* stellt den Basiszugang zu den Datenbanken auf dem MySQL-Server her. Sie bietet eine Oberfläche für SQL-Abfragen. Außerdem unterstützt sie die Programmierung von Stored Procedures.

Die Workbench enthält außerdem die Möglichkeit Datenbanken zu importieren. Die Installation eines MySQL-Servers und einer Workbench reichen damit aus, um Zugriff auf die Daten aus gespeicherten Simulationsläufen zu erhalten und Simulationsläufe mit Parametervariationen durchzuführen.

Strukturierung des Simulationsprogramms

Jede Procedure oder Tabelle wird in einer .SQL-Datei dokumentiert. Zu jeder Procedure oder Tabelle gibt es eine weitere Datei, in der dokumentiert ist, wie der Code getestet worden ist. Zur thematischen Gliederung der Procedures, wird eine Ordnerstruktur errichtet, in der die .sql-Dokumente erstellt und verwaltet werden. Die Ordnerstruktur besteht im Groben aus einem Ordner für die Tabellen und Werte, die zu Beginn der Simulation vorliegen müssen, und einem Ordner für die Ablaufdefinitionen. Innerhalb der Ordner wird nach den Namen der Regeln strukturiert.

Testen: Für die einzelnen Proceduren werden SQL-Abfragen formuliert, die in einem Dokument zum jeweiligen Quelltext gespeichert werden.

Die Versionsverwaltung erfolgt durch Nummerierung der Dateinamen. Alte Dateien werden in Ordner mit dem Vorsatz „history“ verschoben[1]

[1] Ist bei Stored Procedures möglich, weil diese erst in die Datenbank gespielt werden und dann über den Procedurennamen aufgerufen werden. Der Dateiname wird also nicht vom aufrufenden Programm verwendet.

Datenbankverwaltung und Simulationsauswertung: eewcoSimchart

Momentan verwende ich eewcoSimchart für vier Aufgabenbereiche:

1. Starten des MySQL-Servers. (Geht auch über Start > Ausführen > mysqld.exe)

2. Das Aufrufen der pc_monatssequenz zur Durchführung der Simulationsdurchläufe. Mit der Funktion run_pur ist die Simulationszeit deutlich kürzer als beim Aufrufen über pc_run() mit der Workbench. (eewcoSimchart run_pur braucht für die ersten 7 Monate des Modelles „einfaches Geld“ 02:02 Minuten, die Workbench 04:10 und der Aufrudn über die Command-Line überraschende 04:20.)

3. Importieren und Speichern von Datenbanken. (Sollte auch über die Workbench gehen, was ich allerdings nicht eingerichtet bekommen habe.)

4. Anzeigen von Zeitreihen. EewcoSimchart stellt je 5 Zeitreihen in einer Treppenfunktion untereinander dar. Der grafische Überblick über die Verläufe ist für die Analyse von Simulationsdurchläufen mit mehr als 10 Monaten sehr hilfreich.

Das Unterstützungsprogramm wird in *Python* realisiert. Zur Kommunikation mit dem MySQL Server wird das Modul *pymysql* eingesetzt.

Zur Datenbankverwaltung wird auf die MySQL-Programme *mysql* und *mysqld* zurückgegriffen. Da es mir nicht gelungen ist, diese direkt aus Python aufzurufen, werden sie von Python indirekt über *cmd*-Dateien aufgerufen.

Zur Erstellung von Abbildung werden *normierte Procedures* in der Datenbank erstellt, die Datentabellen aus den einprogrammierten Variablen erstellen, wenn sie aufgerufen werden. Aus diesen Datentabellen werden dann die Abbildungen erzeugt. Zur Erstellung der Abbildungen wird das *matplotlib*-Package verwendet. Zur Erstellung der GUI und der Bildschirmausgabe der Abbildungen wird das mit Python mitgelieferte Tkinter eingesetzt. Da matplotlib als Bildausgabeformat nicht die Formate hat, die Tkinter erkennt, wird zur Formattransformation noch das *PIL*-Package benötigt.

Eine Versionsverwaltung von eewcoSimchart erfolgt gegenwärtig nur durch Abschluss eines Release mit Veröffentlichung und Erstellen einer Kopie mit neuer Versionsnummer.

Testen des Gesamtprogramms

Zu jedem Modell wird ein Forum oder Ersatzweise ein Dokument „Modellablauf nachvollziehen“ angelegt. Dort werden auffällige Datenausprägungen erörtert, bis ihr Zustandekommen erklärt worden ist.

Damit wird zum einen ein Weg angelegt, um sich in die Modellstruktur eindenken zu können und andererseits werden so Programmfehler aufgedeckt.

Diskussion

Alle gewählten Softwarebausteine sind im Internet frei verfügbar. MySQL und Python sind etabliert und werden auch für größere Projekte eingesetzt. Es ist deshalb davon auszugehen, dass sie auch in Zukunft verfügbar sein werden und weiterhin betreut werden. Zudem haben sie ihre Zuverlässigkeit unter Beweis gestellt.

Offen ist die Frage der Rechengeschwindigkeit. Es zeichnet sich ab, dass mit neuen Pcs und teiloptimierter SQL die Rechengeschwindigkeit zumindest für Grundmodelle ein flüssiges Arbeiten erlaubt.

Nach einem Strukturcheck ist auch dafür zu sorgen, dass eewcoSimchart nicht nur auf Windows läuft.

Gewählte Struktur des Simulationsprogramms

Grundgerüst

Die Programmierung erfolgt, indem Werte in Tabellen gespeichert und diese mit Übergangsformularen ergänzt oder verändert werden (MySQL-Datenbank, Tabellen und deren Veränderung mit Stored Procedures).

Bei der Gestaltung der Programmstruktur spielen folgende Aspekte eine Rolle:

- Die inhaltliche Zeit- und Koordinationsstruktur des Modells
- Änderbarkeit zwecks Modellentwicklung
- Anforderungen aus Szenariengestaltung oder Veränderungen von Parametern
- Eine besondere Bedeutung kommt der Variationsmöglichkeit der Entscheidungsroutinen zu, da diese am Ende der Begründungkette stehen.
- Test-, Wart- und Optimierbarkeit des Codes

Die Programmstruktur ist momentan wie folgt gewählt:

Es gibt einen Ordner, in dem die Procedures angelegt werden, die die Startaufstellung erzeugen. Der Name für diese Proceduren beginnt mit „pc_zzz_“, damit alle Proceduren, die für die Startaufstellung benötigt werden, in einem Block angezeigt werden. Das sind die Tabellen der Startaufstellung und ihre Daten. Gleichzeitig werden dort auch die Testvorgänge angelegt, mit denen diese Procedures gestestet werden.

Es gibt einen weiteren Ordner Namens „pro_ablaeufe“, in dem die Procedures angelegt werden, die die Abläufe eines Monats beschreiben. Gleichzeitig werden dort auch die Testvorgänge angelegt, mit denen diese Procedures gestestet werden.

Ein Start MySQL-Schema kann angelegt werden, indem nacheinander die Procedures aus dem Ordner „pro_start“ aufgerufen werden. In dieses Schema werden dann auch die Procedures für den Monatsablauf eingelesen.

Es gibt eine Procedure, die die oberste Stufe der zeitlichen Koordination repräsentiert, die Procedure „pc_monatssequenz“. Von dieser Monatssequenz werden Unterprocedures aufgerufen für die einzelnen Themen, wie Arbeitsmarkt, Produktion

etc. Hinzukommen Aktualisierungen der verschiedenen Themen am Monatsanfang und am Monatsende.

Die Verteilung des Modells auf Procedures und Funktionen folgt dabei zwei Grundsätzen:

1. Pro Thema eine eigene Procedurehierarchie.
2. Jede Sinneinheit, für die auch alternative Programmierungswege oder alternative inhaltliche Ausgestaltungen denkbar sind, ist in eine eigene Procedure oder Funktion gegebenenfalls mit Unterprocedures zu fassen.

Zum Arbeiten mit einem Schema gibt es zwei Proceduren: Die Procedure “pc_run“ ruft die Monatssequenz in der gewünschten Anzahl auf. Die Procedure „pc_zzz_startwerte_neu_einspielen“ löscht den Inhalt der Tabellen und ruft die anderen Proceduren auf, die die benötigten Startwerte in die Tabellen schreiben.

Einbindung von Entscheidungsprocedures

Im Ordner „pro_ablaeufe“ wird ein Unterordner angelegt, in dem sich alle möglichen ***Entscheidungsroutinen*** befinden. Für jede Entscheidung wird in diesem Unterordner ein eigener zusätzlicher Unterordner angelegt. Der Procedurename der Entscheidungsprocedures wird zusammengesetzt aus einem gleich bleibenden Teil, der die Procedure als eine Entscheidungsprocedure ausweist und einer Bezeichnung des jeweiligen Entscheidungsalgorithmus (Also etwa: pc_menschen_entscheidungs-eigenschaft_arbeitsangebot__teuer_verkaufen. „pc ist die Präfix für alle procedures, um diese von den Tabellen zu unterscheiden. „menschen_entscheidungseigenschaft_arbeitsangebot“ bezeichnet die Schnittstelle. „teuer_verkaufen“ ist der Name einer Entscheidungsprocedure, die für diese Schnittstelle programmiert worden ist.).

Es wird für jede Entscheidung eine ***Tabelle „[wirtschaftssubjekt]_-entscheidungseigenschaft_[entscheidungsthema]“*** angelegt. (Benennung etwa: menschen_entscheidungseigenschaft_arbeitsangebot). In diese Tabelle wird für die Wirtschaftssubjekte, die die entsprechende Entscheidung zu treffen haben, die Bezeichnung der zu verwendenden Entscheidungsprocedure hineingeschrieben Die Tabelle hat mindestens die Spalten „runde“, „individuums_id“ und „name_des_entscheidungsalgorithmus“.

Für einige Fragestellen kann es aufschlussreich sein, die Entscheidungsproceduren mit Parametern arbeiten zu lassen, die dann mit geringem Aufwand variiert werden können. Dazu kann diese Tabelle „[wirtschaftssubjekt]_entscheidungs-eigenschaft_[entscheidungsthema]“ um weitere Spalten ergänzt werden. Damit die angelegte Tabellenstruktur variabel genutzt werden kann, verwende ich für einen Parameter, den ich verwenden möchte zwei Spalten: Spalte „parameter_1_(INT)“ und eine zweite Spalte „Bezeichnung_p1“.

Wird der zu verwendende Entscheidungsalgorithmus für ein Wirtschaftssubjekt geändert, so wird für dieses Wirtschaftssubjekt eine neue Zeile in dieser Tabelle mit der aktuellen Rundennummer angelegt. Es wird für Wirtschaftssubjekt immer der Entscheidungsalgorithmus mit der höchsten Rundennummer verwendet. Auf diese Weise kann der Entscheidungsalgorithmus in einem Simulationslauf geändert werden und diese Änderung ist auch dokumentiert (wenn auch nur auf Nachschauen sichtbar).

Die Bezeichnung des Entscheidungsalgorithmus wird beim Entstehen des Wirtschaftssubjekts in die entsprechende Tabelle der Entscheidungseigenschaften mit einer entsprechenden Procedure geschrieben.

Um festzulegen, welche Entscheidungsroutine für neue Wirtschaftssubjekte verwendet werden soll, habe ich einstweilen eine weitere Tabelle angelegt: ***pc_[wirtschaftssubjekt]_entscheidungseigenschaft_[entscheidungs-thema]_wert_zu_beginn***.[2] Die Werte dieser Tabelle werden manuell gesetzt. Gegebenenfalls ist eine zusätzliche Procedure notwendig, die die Entscheidungs-eigenschaften von Wirtschaftssubjekten einliest, die Bestandteil der Startaufstellung sind.

Angedacht: Die Tabelle mit den Entscheidungseigenschaften kann auch bei einem Simulationsstopp extern ergänzt und damit die verwendeten Entscheidungsroutinen verändert werden. Alternativ kann eine Procedure am Monatsanfang platziert werden, die automatisch andere Entscheidungsroutinen oder Parameter für bestimmte Wirtschaftssubjekte einschreibt. Auch diese kann mit einer Tabelle versehen werden, wenn die Änderungsparameter während des Simulationslaufs geändert werden sollen. Die Procedure zum Einlesen der Eigenschaft bei Geburt und die dazugehörende Tabelle können ebenfalls so gestaltet werden, dass sich die verwendeten Entscheidungsalgorithmen im Laufe der Zeit ändern. Falls für die Entscheidungs-

[2] Hinweis: Beachte die weitere Procedure zum Einlesen der Startwerte.

routinen wirtschaftssubjektspezifische Parameter benötigt werden, können die Eigenschaftstabellen entsprechend ergänzt werden.

??? Zu klären: mögliche Länge der Bezeichner für die individuellen Entscheidungen. (Gesamtlänge 64 Zeichen abzüglich der Bezeichnung für die Entscheidungsprozedurart = mögliche Länge.)

Schnittstellendokumentation. Da vorgesehen ist, die Entscheidungsroutinen zu wechseln, ist in diesem Fall eine Schnittstellendokumentation besonders wichtig. Die Schnittstelle wird bei der Regel der aufrufenden Procedure angelegt. Die Schnittstelle umfasst die Übergabeparameter. Einstweilen habe ich in den Schnittstellen auch festgehalten, auf welche Informationen die Entscheidungsroutinen zugreifen können. (*Anr. Programmierung.* Vergabe von Zugriffsrechten?)

Eine zweite Schnittstelle für die Eigenschaftsparameter wird direkt im Anschluss an die erste angelegt. Sie enthält die Spalten der beiden Eigenschaftstabellen und die Procedures, in denen die Werte in diese Tabellen eingelesen werden. Die gewünschten Eigenschaftsparameter sind dann in der Dokumentation der jeweiligen Entscheidungsroutine anzugeben und die Einschreibeprocedures, die die Werte in die Eigenschaftstabellen schreiben, entsprechend zu modifizieren.
??? Schnittstellendokumentation im Allgemeinen klären.

Unterstützende Tabellen. Die ein oder andere Entscheidung beruht mehreren Entscheidungsschritten, die zum einen organisiert sein wollen und zum anderen auch dokumentiert werden sollten, um den Programmablauf nachvollziehen zu können. Für die Unternehmensentscheidungen habe ich folgende Organisationsform gewählt:

- Unterstützende Tabellen werden an der thematisch passenden Stelle vorgestellt.
- Die Regeln werden der Verwaltung des Unternehmens zugeordnet.
- Sie werden als unternehmensspezifische Tabelle eingerichtet und bei der Gründung des Unternehmens eingerichtet.
- Sie werden in der Schnittstelle der Entscheidung aufgeführt.

Sammeleintragungen versus Objektdarstellung: Die Unternehmenstabellen

Aus sachlogischer Sicht ist es verlockend, für jedes Objekt eine eigene Tabelle oder zumindest eine eigene Zeile einer Tabelle vorzusehen und Änderungen der Eigenschaften des Objekts quasi am Objekt zu dokumentieren. Aus Gründen der Rechengeschwindigkeit ist dies einstweilen nicht praktikabel und es sind Aggregate zu bilden (Beispiel: Summe der Champignonpackungen). Bei den Unternehmen habe ich den Weg über eigene Tabellen gewählt. Jedes Unternehmen bekommt drei eigene Tabellen zur Arbeits- und Produktionsorganisation. Diese sind wie folgt organisiert:

Wird ein Unternehmen gegründet, werden für dieses Unternehmen bestimmte Tabellen angelegt. Um sie zu identifizieren, beginnen die Tabellenname mit „unternehmen_id_[unid]_“. „[unid]“ steht für die jeweilige Identifikationsnummer des Unternehmens, wie sie im Unternehmensregister eingetragen wird.

Der Bezeichner „unternehmen_id_“ wird zudem gebraucht, um alle dynamisch erstellten Unternehmenstabellen erfassen und für ein erneutes einlesen der Startaufstellung löschen zu können.

Organisation der Procedures für die einzelnen Unternehmen

Für jeden Vorgang, etwa „offene Stellen melden“, wird eine übergeordnete Procedure erstellt, die die Identifikationsnummern der aktiven Unternehmen ausliest. Für diese aktiven Unternehmen wird dann die jeweils zu verwendende Entscheidungsroutine bestimmt und aufgerufen. Die Identifikationsnummer des Unternehmens wird dabei übergeben.

Programminfo-Tabelle

Um das verwendete Modell und vorgenommene Variationen und Szenarioeinstellungen dokumentieren zu können, gibt es die Tabelle „yyy_program_info“. Hier können Hinweise auf das zugrunde liegende Entwicklungsdokument des Modells und ähnliches hinterlegt werden, um nachvollziehen zu können, um welches Programm es sich in der Datenbank handelt.

Die Infotabelle besteht aus den Spalten „art_der_info“, „info“, „bemerkung“ jeweils als text-Spalte. Ich habe als erste Spalten für „art_der_info“ angelegt: „Entwicklungspapier zum Modell“, „Bezugsquelle“, „entwickelt mit MySQL-Version“.

Variationen

Aus der Datenbank nebst Procedures lässt sich die genaue Konfiguration des einprogrammieren Modells nur mit viel Mühe herauslesen. Variationen eines Modells sind daher durch Infodokumente, Ordnerstrukturen für die Programmvariationen und Namensgebung von Datenbanken und Proceduren kenntlich zu machen.

Also etwa:
Das Entwicklungsdokument enthält ein Kapitel über eine bestimmte Variation. Es erläutert den Inhalt der Variation und entwickelt die dazu erforderlichen Regeln. Gegebenenfalls sind weitere Informationen über die Gestaltung des Simulationslaufs zu dokumentieren.

Es wird ein Programmordner für diese Variation angelegt, indem ein Info.txt-File angelegt ist, das kurz die Zuordnung zu einem Hauptmodell nennt. Es enthält einen Hinweis auf das Entwicklungsdokument, in dem die Variation erläutert wird. Es enthält zudem eine Skizze der Besonderheiten dieser Variante.

In dem Ordner befinden sich zudem die Dateien zur Erstellung der zusätzlichen oder geänderten Proceduren, sowie ein Dump des Schemas mit den geänderten oder zusätzlichen Proceduren.

Zeit_abfolge_doku-Tabelle

Bedauerlicherweise gibt MySQL mit den Fehlermeldungen weder die Procedure aus, in der der Fehler aufgetreten ist, noch die Zeile. Das hat zur Folge, dass immer wieder einiger Aufwand vonnöten ist, um die entsprechende Procedure zu identifizieren. Der Aufwand wird noch dadurch erhöht, dass einige Fehler nur in bestimmten Konstellationen auftreten und nicht jede Runde; die Datenbank muss in diesem Fall auf den Stand eines bestimmten Monats gebracht werden.

Eine Möglichkeit die fehlerhafte Procedure zu finden sind ungebundene SELECT-Anweisungen, also SELECT-Anweisungen mit Bildschirmausgaben. Taucht eine bestimmte Bildschirmausgabe auf, dann ist die entsprechende Programmstelle abgearbeitet worden. Der Aufwand dieser SELECT-Anweisungen besteht darin, dass sie wieder auskommentiert werden müssen, nachdem der Fehler beseitigt worden ist. Das Risiko besteht zudem darin, dass die eine oder andere Anweisung übersehen wird und dann etwa den Kommunikationskanal mit dem Python-Programm blockiert, was zu seltsamen Ergebnissen führt.

Ich möchte nun eine andere Möglichkeit versuchen. Anstelle der SELECT-Anweisungen und Bildschirmausgabe schreibe ich nun INSERT-Anweisungen in den Programmablauf. Diese INSERT-Anweisungen schreiben Einträge in eine zeit_abfolge_doku-Tabelle mit den Spalten „runde“, „number“ (INT), „vorgang“ (VARCHAR(64)) und „verwendet_als_marke_von“[3]. Als Zählschritte verwende ich für die Procedures der „pc_monatssequenz“ 10.000er Schritte. Die für die Einrichtung benötigten Anweisungen finden sich m Ordner ablaufe/zeit. Eine Zeile für die Leerung der Tabelle ist in die Procedure „pc_zzz_startwerte_neu_einspielen“ eingefügt.

Für das Debuggen scheint mir das deutlich bequemer. Es kostet allerdings Performance. Auf meinem etwas langsamen System etwa 1 Sekunde für 12 INSERTS. Das summiert sich auf etwa 3 Sekunden pro Monatsdurchlauf. Wobei die Schätzung möglicherweise auch völlig falsch liegt, da ich einen Einzelaufruf in einer Testprocedure nur für das INSERT-Statement simuliert habe.

Möglicherweise lässt sich das Verfahren etwas beschleunigen, in dem die INSERT-Anweisung in eine IF-Abfrage eingebunden wird, die den Wert einer @debug-Variablen vergleicht. Wenn der Wert = 0 ist, dann wird kein INSERT-vorgenommen.

Sollte die Performance an der Größe der yyy_debugging_table geknüpft sein, ließe sich die Tabelle auch zu Ende eines Monatsdurchlaufs wieder leeren.

[3] Die Tabelle wird zudem für das Modell verwendet, wenn für eine Procedure ermittelt werden muss, ob sich vor oder nach einem anderen Vorgang aufgerufen wird.

Quellen für die Software

MySQL 5.5

http://www.mysql.de/downloads/mysql/

Workbench

http://www.mysql.de/downloads/workbench/

Die Simulationsprogramme

www.eewco-rumpelt.de > T4 ökonomische Theorie

Python 2.6
(Version 2.6 solange dies der größte gemeinsame Versionsnenner für die benötigten Packages ist)

http://www.python.org/download/releases/2.6.6/

(Es gibt eine Version 2.6.7. Allerdings werden dort keine Windows- und Macinstaller getrennt geführt. Ich weiß nicht, wie der angebotene Download dann zu installieren ist.)

pymysql
(Eine Alternative dazu ist mysqldb. Hab ich mir allerdings schwerer getan zu installieren. Der Code dürfte auch mit mysqldb funktionieren, wenn die import-Anweisungen in den eewcoSimchart-Modulen auf mysqldb umgestellt werden.)

http://pypi.python.org/pypi/PyMySQL/0.2

matplotlib

(Die Version, mit der eewcoSimChart erstellt worden ist, ist 1.0.0 für py 2.6. Denke aber, dass es keine Probleme gibt, die neuere Version zu verwenden. Die neue Version hat den entscheidenden Vorteil, dass sie erlaubt mehrmals show() aufzurufen und so mehrere Abbildungen gleichzeitig anzuzeigen. Könnte für die Weiterentwicklung interessant sein.)

http://sourceforge.net/projects/matplotlib/files/matplotlib/matplotlib-1.0.1/

PIL (Python Imaging Library)

http://www.pythonware.com/products/pil/

Unzipper für die .eggs. Unter Windows vorhanden.

CMD
unter Windows vorhanden: Start > ausführen > cmd

eewcoSimchart

www.eewco-rumpelt.de > T5 Simulationsumgebung

Betriebssysteminfo

Das Programm wurde auf einem Rechner mit Windows XP, Nutzeraccount erstellt.

eewcoSimchart erfordert das Schreiben von zwei Dateien in den Ordner, in dem sich die eewcoSimchart-Programmdateien befinden.

eewcoSimchart verwendet das Modul os und das Modul shutil. Diese laufen nur auf Windows und ich meine Unix-Systemen. Eine Kapselung der betriebssystemspezifischen Funktionen und die Bereitstellung einer Linuxversion stehen auf der Wunschliste.

Auch das Package PIL gibt es nur für Windows. Hier ist eine alternative, plattformunabhängige Bildformatübertragung von png in Tkinterformat gesucht.

Installation der Programmkomponenten

Im Folgenden das Protokoll einer Installation auf Windows Vista, Version 6.0.

Die Basis: MySQL, Workbench und ein eewco-Simulationsprogramm

Ordner anlegen und Downloads durchführen
Als Ort für die Modellarbeit lege ich in meinem Benutzerkonto einen Ordner „eewco modell einfaches geld“ an.

Dann besorge ich den Ordner mit den Programmdateien für das Modell „einfaches Geld“ („t4-code-reihe-geld-1-einfach-geld-[6]“) und lege ihn in dem eben erstellten Ordner ab. Der Ordner wird als .zip-Datei auf eewco.rumpelt.de > t4 ökonomische Theorie > Reihe Geldtheorie I > einfaches Geld zur Verfügung gestellt.

Aus der gleichen Quelle werden auch die beiden Dokumente der Modellbeschreibung in diesen Ordner kopiert:

t4-reihe-geld-1-einfaches-geld-entwicklungspapier-[6]r.pdf und

t4-reihe-geld-1-einfaches-geld-entwicklungspapier-[6]-regelzusammenstellung-[1]r.pdf.

Einen weiteren Ordner „Programmdownloads“ erstelle ich in meinem Konto für die Programme der Simulationsumgebung. Die extern heruntergeladene Software möchte ich hierhinein downloaden und von hier installieren.

Ich surfe zu der im Abschnitt „Quellen für die Software“ angegebenen Seite für MySQL. Dort finde ich fünf Installer für einen MySQL Community Server 5.5.20. Ich meine mich zu erinnern, dass die beiden MSI Installer diejenigen sind, die ich mit meinem Know-how anwenden kann. Es gibt einen für 32Bit und einen für 64 Bit. Ich schaue nach unter Start > Einstellungen > Systemsteuerung > System und finde, dass der PC ein 64Bit System hat und lade den Installer über einen http-Mirror herunter. (Wobei ich mich frage, warum der „Speichern“-Button unscheinbar am unteren Bildschirmrand auftaucht. Ich wähle den Weg über „Speichern unter“.)

Auf der Workbench-Seite gibt es die MySQL Workbench 5.2.37. Es gibt einen MSI-Installer. Allerdings in der 32Bit-Version. Eine kurze Recherche gibt Anlass zu der Hoffung, dass Windows intern für 32Bit Programme einen Adapter bereitstellt. Ich lade also den 32Bit Installer herunter.

MySQL installieren
Nächster Schritt: Installation von MySQL. Doppelklick auf die exe-Datei. „Typical“ Installation gewählt. Am Ende den „Launch Configuration Wizard“ aktiviert gelassen.

Configuration Wizard:

1. Detailed Configuration
2. Server machine (medium memory usage) (Überlegung: Die Berechnungen brauchen einiges an Rechenleistung, aber ich möchte auch andere Programme derweil geöffnet haben, die auch Ressourcen brauchen)
3. Mulitfunctional Database
4. Default tablespace gelassen. (Der Benutzer braucht Schreibrechte auf diesem Pfad, sonst einen entsprechenden wählen)
5. Concurrent Connections: Manual 5.

6. Enable TCP/IP gelassen. Enable Strict Mode gelassen (Wichtiger Punkt für die Programmierung, damit Programmfehler schneller entdeckt werden. Hat allerdings Seiteneffekte.)
7. Standard Character Set gelassen.
8. Install as Windows Service. Lauch MySQL-Server automatically disabeled – Der Rechner wird nur selten für die eewco-Programmierung genutzt. Da muss der Server nicht immer mitlaufen.
   Include Bin Directory in Windows PATH. Yes, enable. Könnte die eine oder andere Pfadangabe vereinfachen.
9. Securitiy Settings. Ein root.

Jetzt die Workbench installieren. Doppelklick auf die exe-Datei. Ratter ratter. Dann Meldung: Visual C++ 2010 Redistributable Package fehlt noch. Nach OK-klicken poppt wieder der Installations-Wizard auf. Links unten ein Button zu den Prerequisites. Auf der Internetseite, die sich öffnet, wieder den Prerequisites folgen. Da lässt sich dann das gewünschte Paket downloaden (Wobei zwischen 32 und 64Bit nicht unterschieden wird). Das Paket speichert sich irgendwo in Downloads. Ich finde unten links eine Frage nach „Behalten oder Verwerfen“ und finde darüber den Download. Den ziehe ich in den angelegten Ordner „Programmdownloads“ und installiere das Package von da. Anschießend funktioniert auch die Installation von Workbench. Ich wähle die complete Variante.

MySQL-Server starten.
Erster Versuch
Start > Ausführen: „mysqld --console“ (Geht nur, wenn in der Configuration von MySQL mysql als Windows-PATH eingetragen worden ist. Sonst in den Programmordner von MySQL klicken und dort den Ordner „bin“ suchen. Dort ist mysqld gespeichert.)
CMD Fenster öffnet sich wie erwartet. Allerdings mit der Fehlermeldung
InnoDB: Operating system error number 32 in file operation …
Google Suche.

1. Fund: If so, make sure you have write permission to "c:\mysql\data"?
   Könnte sein, dass da das Problem liegt. Ich durfte unter Programme auch nichts speichern.
2. Fund: Der Server war bei mir schon am Laufen.

Datenbankserver läuft schon
Ich schaue nach bei Start > Programme > MySQL > MySQL 5.5 > Command Line Client. Tatsächlich, der öffnet sich. Fragt nach dem Passwort. Danach kann ich \h eingeben oder „USE information_schema;“ und „SHOW TABLES;“. Sieht so aus, als wenn der Server schon läuft.

Den Datenbankserver starten, wenn er nicht läuft
Nach einem Neustart muss die Datenbank manuell gestartet werden, jedenfalls mit der oben vorgeschlagenen Einstellung. Ich nutze dafür den folgenden Weg:
Start > Ausführen. Dann „Durchsuchen“ klicken und bis zur Datei mysqld.exe vorarbeiten. Den Pfad zu dieser Datei mit „öffnen“ in die Befehlszeile laden. Mysqld.exe befindet sich typischerweise im Verzeichnis C:\Programme\MySQL\ MySQL 5.5\bin. Hinter die Pfadangabe in der Befehlszeile noch „ --console“ eingeben.

Das sorgt dafür, dass das CMD-Fenster geöffnet bleibt. „OK“ klicken und der Datenbankserver wird gestartet.

Starten der Workbench
Ein Icon finde ich nicht auf dem Desktop, aber einen Eintrag unter Start > Programme > MySQL. Workbench öffnet sich.
Ich klicke in der linken Spalte unter SQL-Development auf die blau unterlegte „Local instance MySQL“. Ich werde wieder nach den root Passwort gefragt und bin dann auf der SQL-Entwicklungsoberfläche. Links entdecke ich eine Datenbank mit dem Namen „test“.

Import der ersten Modelldatenbank
Aus welchen Gründen auch immer, klappt mal das eine mal das andere Verfahren nicht. Ich beschreibe im Folgenden 2 Wege, die zum Erfolg geführt haben.

Die benötigte Datenbank-Dump befindet sich in dem eingangs angelegten Ordner „eewco modell einfaches geld“. Dort in dem Ordner mit den Programmdateien „t4-code-reihe-geld-1-einfach-geld-[6]“ gibt es einen Ordner „Schemata“. In diesem Order sind die Datenbank-Dumps zu dem Modell. Die Dumps haben die Dateiendung *.sql.

1. Weg: Installation mit der Workbench
Ich starte die Workbench. Im Bereich „Serveradministration“ klicke ich auf „Local MySQL“. Es öffnet sich ein Fester „Admin (Local MySQL)“. Im linken Abschnitt gibt es unten eine Rubrik DATA EXPORT/RESTORE und dort einen Link „DATA Import/Restore. Den klicke ich an. Es öffnet sich ein Fenster „Import from Disk“. Dort werden 2 Optionen zur Auswahl gestellt. Ich wähle „Import from Self-Contained File“. Für den File-Path klicke ich mich zu dem Datenbank-Dump durch und wähle diesen. Ganz unten lässt sich dann der Import starten.

(Genau genommen habe ich noch einen Zwischenschritt eingelegt: Auf der Startseite der Workbench in der Rubrik „SQL Development“ habe ich auf „Local instance MySQL“ geklickt. Es öffnet sich ein „SQL Editor“-Fenster. Im Query Fenster habe ich eingegeben: „DROP DATABASE einfaches_geld_grs;“ Und dann durch Klick auf den Blitz mit dem Cursor ausführen lassen. Anschließend die Anweisung „CREATE DATABASE einfaches_geld_grs;“ eingetippt und ausgeführt. Überprüft im linken Fenster SCHEMAS. Dort sind rechts zwei runde Pfeile, mit denen der Stand aktualisiert werden kann. Nach dem Aktualisieren wird die Datenbank „einfaches_geld_grs“ aufgeführt. Diesen Zwischenschritt vorausgesetzt habe ich dann im „Import from Disk“-Fenster in das Feld „Default Schema to be imported to“ „einfaches_geld_grs“ eingetragen“.)

2. Weg: Installation mit mysql.exe und CMD
Jetzt brauche ich nur noch die Modelldatenbank auf dem Server. Ich klicke auf die rechte Spalte Serveradministration der Homeseite der Workbench (zu erreichen über das Häuschen links oben), und klicke auf das blau hinterlegte „Local MySQL“. Nach Passwortabfrage öffnet sich die Administrationsseite.

Der Versuch die Datenbank aus dem Ordner „eewco modell einfaches geld/schemata“ zu importieren führt zu der Fehlermeldung „Access denied“. Aha. Wohl dieses Datenschreibzugriffrechteproblem. In der linken Spalte findet sich der Menüpunkt „Options File“. Unter dem Reiter „General“ gibt es einen Eintrag „data dir“.

Hier wähle ich nun einen Ordner aus, bei dem ich sicher bin, dass ich Schreibrechte darauf habe. Der Erfolg der Maßnahme wird bei einem erneuten Importversuch deutlich: Die Fehlermeldung lautet nun anders.

Ich beschließe den Fußweg einzuschlagen. Der geht so: Start > Ausführen: „CMD“ eingeben und auf OK klicken. Es öffnet sich die CMD-Console. Dort mysql eingeben. Wenn nun die Fehlermeldung „Access denied for user ODBC…“ erscheint, sehr gut, dann hat Windows das mysql-Programm gefunden. (Wenn nicht dann über „cd..“, „DIR“, „cd Programme“ und ähnlichen Anweisungen den Ausgangspfad des Prompts zu MySQL\MySQL Server 5.5\bin verlegen.)

Nun ließe sich die Datenbank über das Programm mysql importieren, wenn es nicht einen Bug gäbe. Ich finde im Netz:

This is a **known bug** at MySQL.

http://bugs.mysql.com/bug.php?id=42996http://bugs.mysql.com/bug.php?id=40477

As you can see this has been a known issue since 2008 and they have not fixed it yet!!!

WORK AROUND
You first need to create the database to import. It doesn't need any tables. Then you can import your database.

1. first start your MySQL command line (apply username and password if you need to)
   C:>mysql -u user -p

2. Create your database and exit

   mysql> DROP DATABASE database
   mysql> CREATE DATABASE database
   mysql> Exit

3. Import your selected database from the dump file
   C:>mysql -u user -p -h localhost -D database -o < dumpfile.sql

You can replace localhost with an IP or domain for any MySQL server you want to import to. The reason for the DROP command in the mysql prompt is to be sure we start with an empty and clean database.

// Max

Gut. Also Start > Programme > MySQL> MySQL Server 5.5>Command Line. Da hier noch kein Importversuch unternommen worden ist, verzichte ich auf das DROP DATABASE-Statement und gebe ein: „CREATE DATABASE einfaches_geld;“. Return drücken dann „Exit;“ eingeben und returnen um die Command Line wieder zu beenden.

Weiter geht’s wieder mit dem CMD-Fenster. Um die manuelle Pfadangabe zu vereinfachen, habe ich den Datenbankdump „einfaches_geld_m0“ nocheinmal direkt in das C:-Verzeichnis kopiert. Dann sollte der folgende Befehl im CMD-Fenster zum Import führen:

„mysql –u root pPasswort einfaches_geld < C:\einfaches_geld_m0.sql“.

Beachte: Kein Leerzeichen zwischen dem p und dem Passwort. „einfaches_geld“ ist der Name der Datenbank, die wir oben eingerichtet haben.

Return-Taste drücken. Im CMD-Fenster sieht es nun so aus, als sei nichts passiert. Ist das Workbench-Fenster SQL-Editor noch offen, ist auch dort nichts passiert. Der Stand muss aktualisiert werden. Der der Kopfzeile des linken Fensters mit den Datenbanken gibt es dazu Pfeile im Kreis. Diese drücke ich. Und siehe da: die Datenbank ist vorhanden!

(
Ein bisschen in der Datenbank stöbern

Zuerst muss die Datenbank ausgewählt werden, auf die sich die folgenden Abfragen beziehen. Dazu im rechten Fenster, das zu Beginn den Namen „SQL-File 1“ trägt, eintippen: „USE einfaches_geld;“. Dann den Cursor in diese Zeile setzen. In der Iconleiste des Fensters befinden sich zwei gelbe Blitze. Der linke lässt alle Befehle des Skripts ausführen. Der rechte Blitz mit dem Cursor drauf nur den Befehl, auf dem den Cursor steht. Den anklicken. Links müsste nun der Datenbankname fett geworden sein.

Den Inhalt der Tabellen ansehen geht so. Nehmen wir als Beispiel die Tabelle „geld_konten“: „SELECT * FROM geld_konten;“ und den Blitz betätigen. Das Ergebnis der Abfrage wird in dem Feld unter dem Skriptbereich angezeigt. Je nach importierter Datenbank erscheint jetzt ein Tabellenkopf mit oder ohne Werte.

Das importierte Modell laufen zu lassen geht so: Es gibt oben eine Reiterleiste mit dem Reiter „File“. Dort gibt es den Menüpunkt „Open SQL-Script“. In dem Auswahlfenster ist nun der Ordner mit dem Modell zu aufzusuchen. In diesem Ordner befindet sich ein Unterordner „programm_ablaeufe“. Dort findet sich ein Skript Namens „aaa_runstatements [1].sql“. Dieses öffnen. Mit dem Befehl „CALL pc_run(1)“ wird die pc_monatssequenz sooft aufgerufen, wie in der Klammer angegeben.
)

EewcoSimchart

EewcoSimchart ist in Python programmiert und benötigt einige Erweiterungen zurück: Eine für die Darstellung von Funktionen (matplotlib), eine, um mit MySQL zu kommunizieren (pymysql) und eine, um die Abbildungen anzeigen und speichern zu können (PIL).

Versionscheck

Alle diese Packages müssen dieselbe Version von Python unterstützen. Deshalb ist zunächst zu prüfen, für welche Python-Version die Packages vorliegen.

Matplotlib: Gibt’s mittlerweile auch für 2.7.

Pymysql: Es gibt im Internet auch Spuren einer Version für 2.7 und 3. Den entsprechenden Download habe ich aber nicht finden können.

PIL: Versionen bis 2.7.

Also bleibt es bei Python 2.6.6.

Download

Python: Ich nehme den Windows x86 MSI Installer (2.6.6)(sig).

Pymysql: PyMySQL-0.3_1-py2.6.egg (md5)
Matplotlib: matplotlib-1.0.1_r0-py2.6-win-32.egg
(Nachsehen unter Start > Einstellungen > Systemsteuerung > System sagt mir, dass der Rechner einen AMD-Prozessor hat. Deshalb habe ich in einem ersten Versuch die amd64 Version installiert. Daraus ergab sich die Fehlermeldung „… %1 keine Win32 Anwendung“. Mit der nun gewählten Variante bleibt die Fehlermeldung aus.)
PIL: Python Imaging Library 1.1.7 for Python 2.6 (Windows only)

Installation
Los geht’s mit der Installation von Python. Doppelklick und Voreinstellungen übernommen. Nach der Installation sehe ich nach, ob die Installation auch erfolgreich gewesen ist. Ich öffne die Python Shell (Start > Programme > Python 2.6 > IDLE) und tippe just vor fun ein: >>> a = ‘hallo‘, (Hochkomma über #) Enter. >>> print a, Enter. Funktioniert. Dann gebe ich zum Reaktionstest für die nächsten Schritte ein: >>> import pymsql, Enter. Meldung: ImportError: No module named pymysql.[4]

Pymysql und Matplotlib liegen als ZIP-Ordner vor. Ich kann es nicht erkennen, aber es kann sein, dass sie die Endung .egg haben. Zu den Python-eggs habe ich im Netz Folgendes gefunden:

> Was du auf jeden Fall machen kannst ist das Egg oder die EXE zu entpacken. Das Egg ist eigentlich ein umbenanntes Zip-Archiv und die EXE-Datei ist ein Zip-Archiv mit eigenem Entpacker von dem her ist es nicht schwer das einfach zu entpacken und den darin enthaltenen MySQLdb-Ordner in site-packages deiner Python-Version zu verschieben.
> Quelle: http://www.python-forum.de/viewtopic.php?f=2&t=18746

Ich entpacke pymysql indem ich mit der rechten Maustaste auf den Zip-Ordner klicke und im Kontextmenü „Alle extrahieren“ wähle. (Wenn sich mit der rechten Maustaste kein Zip Programm öffnet, dann zum Beispiel 7zip von der CHIP-Seite herunterladen und installieren.) Es wird ein Order PyMySQL-0.3_1-py2.6 angelegt, indem sich ein Ordner pymysql befindet. Den kopiere ich und füge ihn unter C:\Python26\Lib\site-packages ein. Dann teste ich ob die Installation erfolgreich war und gebe in der Python-IDLE ein: >>>import pymysql. Reaktion: >>>. Diesmal keine Fehlermeldung – es hat also funktioniert.

Das Gleiche mit matplotlib führt allerdings nur zu einem Teilerfolg. Für matplotlib fehlt noch das Package numpy:

> http://matplotlib.sourceforge.net/users/installing.html#installing-on-windows
>
> For standard python installations, you will also need to install numpy in addition to the matplotlib installer. On some systems you will also need to download msvcp71.dll library, which you can download fromhttp://www.dll-files.com/dllindex/dll-files.shtml?msvcp71 or other sites. You will need to unzip the archive and drag the dll intoc:windowssystem32.

[4] Auf meinem Laptop ist ein Python 2.2 vorinstalliert, wird allerdings bei Software nicht aufgeführt, so dass ich die Version auch nicht deinstallieren konnte. Des Rätsels Lösung lag in dem Umschreiben der Umgebungsvariablen PYTHONPATH, TCL_LIBRARY und TK_LIBRARY.

Quelle: http://matplotlib.sourceforge.net/users/installing.html#installing-on-windows

Numpy für Python gibt's auf:

http://sourceforge.net/projects/numpy/files/

Ich lade numpy-1.6.1-win32-superpack-python2.6.exe herunter.
(Auf der Seite
http://www.scipy.org/Download#head-f64942d62faddeb27278a2c735e81ef2a7349db0
gibt es eine inoffizielle Version für 64 Bit. Ich versuchs mal mit der 32Bit Version.)
Die Numpy-Datei ist eine .exe-Datei. Ich doppelklicke und starte die Installation. Installieren möchte sich das Programm in C:\Python26\Lib\site-packages, was konsistent ist. Nach der Installation führt die Eingabe von >>>import matplotlib zu keiner Fehlermeldung mehr.

PIL installiert sich wie numpy.

Damit wären die Prerequisites für eewcoSimchart hergestellt.

eewcoSimchart

Nach dem Download von „eewcooSim [1p0].7z" möchte die den Ordner wie gewohnt über den Klick mit der rechten Maustaste und „Alle extrahieren" entpacken, aber das Betriebssystem des Computers kennt das Format 7z nicht.

Auf
http://www.chip.de/downloads/c1_downloads_hs_getfile_v1_16093343.html?t=1327921704&v=3600&s=a2b144d53af0f8e6bb7b987961e62501
finde ich eine entsprechende Software. Installieren durch Doppelklick.

Zip-Programm starten über Start > Programme > 7-zip > 7-zip File manager. Dann den 7z Ordner über >Datei > Öffnen aufrufen. Wenn der Ordner im Anzeigenfenster auftaucht. das dicke blaue Minus für entpacken klicken, fertig.

Den entpackten eewcosimchart-Ordner verschiebe ich an eine Stelle, zu der ich Schreibrechte besitze, weil eewcosimchart temporäre Bildordner und die Einstellungen speichert.

eewcoSimchart einrichten.
Ich starte die Python Shell (Start > Programme > Python 2.6 > IDLE). Ich gehe auf >File > Open und klicke mich zum eewcoSimchart-Ordner durch. Dieser Ordner enthält einen Ordner namens eewcosimchart-python-[3] mit den Programmdateien.

Die Startdatei heißt eewcosimchart_start.py.

Die öffne ich. Mit > Run > Run Module oder drücken von F5 wird eewcoSimchart gestartet – wenn es funktioniert.

Es fehlt für matplotlib noch das Modul „dateutil“. Eine Ordner dateutil befindet sich bei den entpackten Ordnern von matplotilib. Den Ordner dateutil ebenfalls in site-packages verschieben.
(oder Download von
http://labix.org/python-dateutil#head-42a94eedcff96da7fb1f77096b5a3b519c859ba9
Datei: python-dateutil-1.5.tar.gz.
2x entpacken mit dem oben installierten Packprogramm für 7z. Durchklicken bis ein Ordner dateutil auftaucht und den wie gehabt in den site-packages-Ordner kopieren.)

Ein erneuter Startversuch von eewcoSimchart und das Programm öffnet sich. Als erstes lege ich den Ordner für die temporären Bilddateien fest: toolbar > Schema-Management > tempchart-Dir > Pfad aendern und einen gültigen Pfad wählen, für den Schreibrechte vorhanden sind.

Als nächstes den MySQL-Server starten: toolbar > Schema-Management > mysqld starten. Der angegebene Pfad entspricht den Voreinstellungen bei der Installation von MySQL (C:\Programme\MySQL\MySQL Server 5.5\bin). Bei Eintrag als Windows-PATH Variable Pfadangabe wohl unerheblich.

Auf dem Rechner, auf dem ich eewcoSimchart entworfen habe, öffnet sich durch das Starten ein cmd-Fenster und es meldet sich die Firewall. Beides geschieht nun nicht. Ich überprüfe, ob der MySQL-Server läuft: Start > Programme > MySQL > MySQL Server 5.5> Command Line. Passwort eingeben und werde begrüßt. Der MySQL-Server läuft also.

Verbindungsdaten von eewcoSimchart zum Server einrichten. Die Verbindungsdaten befinden sich in der Datei connection.py relativ am Anfang:
Die Verbindungsdaten sind zu aktualisieren, insbesondere das Passwort. Als Programm kommt dafür neben der IDLE von Python auch der Windows Editor in betracht:

```
# andere Verbindungsdaten initialisieren
self.host = 'localhost'
self.user = 'root'
self.passwd = 'passwort'
self.port = 3306
```

Jetzt fehlt eewcoSimchart noch der Name einer Datenbank, die auf dem Server läuft. Der Datenbankname wird gespeichert in der Datei „einstellungen.data“. Diese Datei mit dem Windows Editor öffnen (Computer bzw. Arbeitsplatz, rechte Maustaste, öffnen mit … Nicht mit Winword öffnen, sonst werden Formatinformationen hineingeschrieben, die eewcoSimchart die Orientierung nehmen). Es gibt eine Zeile „current_db: Datenbankname“. Für den Datenbankname den Namen der Datenbank hineinschreiben, die oben eingerichtet worden ist. Wichtig ist, dass es 1 Leerzeichen zwischen „current_db:“ und dem Datenbanknamen gibt.

Nochmal die Oberfläche von eewcoSimchart schließen und mit Run > Run Module oder F5 neu starten und jetzt funktioniert’s!

Erste Eindrücke eines eewco-Simulationsprogramms mit der Workbench erhalten

Voraussetzung: MySQL und Workbench installiert. Der MySQL-Server läuft (mysqld.exe gestartet, s.o.). Die Verbindung von der Workbench zum Server funktioniert und es wurde ein Datenbank mit einem eewco-Modell importiert.

Nun können die unterschiedlichen Tabellen mit SQL-Abfragen betrachtet werden. Ich möchte besonders auf eine .SQL-Datei mit dem Namen „aaa_runstatements_[1].sql“ aufmerksam machen. Sie befindet sich im Ordner „programm_ablaeufe“. Diese Datei enthält drei CALLs:

1. CALL pc_run(1). Die Procedure „pc_run“ ruft die Procedure „pc_zeit_monatssequenz“ auf, und zwar sooft wie in der Klammer angegeben wird.

2. CALL pc_zzz_startwerte_einspielen(). Diese Procedure bringt die Datenbank wieder in den Ausgangszustand in Monat 0.

3. SELECT * FROM zeit_ablauf_doku. Sollte das Programm aufgrund eines Fehlers abbrechen, lässt sich über die zeit_ablauf_doku-Tabelle herausfinden, wie weit das Programm gekommen ist. Den letzten Eintrag dieser Tabelle in der Prodedure pc_zeit_monatssequenz suchen. Weitere Einträge werden auch von einigen untergeordneten Proceduren vorgenommen, etwa der Procedure pc_arbeitsmarkt.

Struktur der Modellprogrammierung: Beispiel

Die inhaltliche Arbeit, die der Programmierung in wesentlichen Teilen vorausgeht, wird in einem Entwicklungsdokument vorangetrieben. Als Beispiel möchte ich hier das Dokument „???[Entwicklungspapier, Reihe: Geldtheorie I, Ausgangsmodell]“ heranziehen. Hier werden zunächst die inhaltlichen Fragen gestellt, mögliche relevante Aspekte gesammelt, ausgewählt und schließlich in eine zeitliche Abfolge gefügt. Die Aufteilung der Abfolge auf einzelne Schritte definiert die spätere Programmstruktur.

Das Programm an sich besteht aus Datentabellen und Proceduren. Die Proceduren berechnen auf der Grundlage der vorhandenen Daten die Werte für den aktuellen Modellmonat und fügen sie in die Tabellen ein. Die oberste inhaltliche Procedure ist eine Procedure names pc_monatssequenz. Sie ruft die weiteren Procedures auf, die notwendig sind, um das Geschehen innerhalb des Monats abzubilden.

Zu Beginn ist zunächst eine Datenbank anzulegen und die Tabellen einzurichten, auf die die Proceduren dann zugreifen können. Außerdem sind eventuelle Startwerte in die Tabellen zu schreiben. Die dafür notwendigen SQL-Befehle werden in Skripten verwaltet, die sich im Ordner pro_start befinden.
Um in der Datenbank Procedures für den Durchlauf von Procedures zum Einspielen von Startwerten unterscheiden zu können, werden die Procedures für die Startwerte mit einem vorangestellten zzz gekennzeichnet.
Für jede Tabelle gibt es ein Dokument, das die Anweisung zur Erstellung der Tabelle enthält, ein Dokument, das die Anweisungen für das Einlesen der Startwerte enthält und ein Dokument das Anweisungen enthält, die verwendet worden sind, um die Funktionsfähigkeit der beiden ersten zu testen.
Der Startordner enthält zudem eine Procedure zzz_pc_startwerte_neu_einspielen, die so eingerichtet ist, dass in alle Tabellen die Startwerte geschrieben werden, bzw. andere Werte gelöscht werden.

Die Beschreibung der Monatssequenz befindet sich im Ordner pro_ablauf. Der Ablauf ist auf eine ganze Reihe von Proceduren verteilt. Für jede Procedure gibt es ein Zusatzdokument, das Anweisungen enthält, mit denen getestet worden ist, ob die Procedure ihre Funktion erfüllt.

Zu Testzwecken gibt es außerdem ein Dokument Namens aaa_run.sql, mit dem eine Procedur names pc_run aufgerufen werden kann. Diese Procedure ruft die pc_monatssequenz so oft wie gewünscht auf.

Die Dokumente tragen im Namen eine eckige Klammer mit Nummer. Dies ist die Versionsangabe für das jeweilige Dokument. Momentan nummeriere ich jedes Dokument separat.

Programmcode weiterverwenden, Lizenzanpassungen

Der Programmcode kann gemäß GPL-Lizenz weiterverwendet, verändert und wieder unter GPL-Lizenz veröffentlicht werden. Dazu sind die Änderungen kenntlich zu machen, und der Programmname so zu ändern, dass erkenntlich ist, dass es sich um ein anderes Programm oder eine andere Programmversion handelt.

Auch die Lizenzbestimmungen sind zu aktualisieren und zwar um den Autor, der nun auch ein Copyright hat, mit Datum und Programmname.

??? Wie die Vorgehensweise bei unverändert übernommenen Programmskripten laut GPL sein müsste, habe ich noch nicht nachgeforscht.

Da das Programm im vorliegenden Fall kein Anwendungsprogramm sondern ein Forschungswerkzeug ist, ist mit vielen Änderungen von zahlreichen Autoren zu rechnen. Mit dem folgenden Vorgehen versuche ich den Aufwand der Lizenzierung möglichst gering und doch vollständig zu halten und auch den benötigten Platz in den einzelnen Skripten überschaubar zu halten:

Es gibt eine Versionsdatei namens „license_version_document_current.txt“ mit folgendem Inhalt:

> current program name (referred to as “the program”):
> [program name]
>
> The program is part of the eewcoSim-framework to analyse economic coordination.
> The other listet programs are part of the program.
>
> The program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, version 3.
>
> The program is distributed in the hope that it will be useful,
> but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
> See the GNU General Public License for more details.
>
> You should have received a copy of the GNU General Public License along with the program.
> If not, see <http://www.gnu.org/licenses/>.
>
> Copyright
> program name:
> date:
> author:
> contact:
> changes:
> documentation source:
> root_0: current

In den einzelnen Skripten wird dann auf dieses Dokument verwiesen. Und zwar in der folgenden Form:

```
#   Copyright [year] [name]
#   contact: [some adress]
#   This file is part of the eewcoSim-framework to analyse economic coordination.
#   It was built as part of the model:
#   [model name]
#   It is now part of the above named models and
#   of all models documented in the files
#   “license_version_document_current” and
#   “license_version_document_root_[number]”
#   building a line starting from the above named models and the latest model.

#   The program of „[model name]“
#   is free software: you can redistribute it and/or modify
#   it under the terms of the GNU General Public License as published by
#   the Free Software Foundation, version 3.

#   The program of „[model name]“
#   is distributed in the hope that it will be useful,
#   but WITHOUT ANY WARRANTY; without even the implied warranty of
#   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
#   GNU General Public License for more details.

#   You should have received a copy of the GNU General Public License
#   along with the program of
#   „[model name]“.
#   If not, see <http://www.gnu.org/licenses/>.
```

Durch den Verweis vom Skript auf das license_version_document_current wird bestimmt, zu welchen Programmversionen ein Skript gehört. Es wird zudem nachvollziehbar, wer das Skript erstellt oder geändert hat und wo Dokumentationen dazu erhältlich sind.

Wird nun ein Programm geändert, dann wird das Copyright der veränderten Skripte ergänzt (Jahr, Autor, Programmname, Hinweis auf vorgenommene Änderungen) und die Datei license_version_document_current erhält einen neuen Eintrag und einen neuen aktuellen Programmnamen.

Wird ein neues Programm aus mehr als einem bestehenden Programm abgeleitet, so werden die license_version_document_current-Dateien der eingegliederten Programme in licence_version_document_root_[number] umbenannt und in den gleichen Ordner zu der aktuellen licence_version_document_current-Datei kopiert. Dort werden die root-Dokumente im Copyrightblock aufgeführt.

eewcoSimchart verwenden

Die folgende Dokumentation ist einen für den Anwender gedacht, der in einem bestehenden eecwo-Simulationsprogramm stöbern und die Funktionalität von eecwoSimchart dafür einsetzen möchte. Die Dokumentation möchte auch die hinter den Funktionen liegende Programmstruktur beschreiben, so dass es einfacher wird, das Programm nach eigenen Ideen zu erweitern oder umzugestalten.

Start

1. Möglichkeit. Die Python-Shell starten (IDLE). Dann die Datei eewcosimchart_start.py im Ordern eewcosimchart [1] öffnen und dort aus der Menüleiste oben „Run“ und dann „Run Module“ auswählen. Der Vorteil dieser Vorgehensweise: Etwaige Fehlermeldungen können in der Shell gut gelesen werden und das Werkzeug zu ihrer Bearbeitung ist auch offen.

Nachteil: Auch die Pythton-Shell arbeitet mit tkinter als GUI wie eewcosimchart auch. Das soll sich beißen können. Was ich beobachten kann, ist das die Programmausführung nicht ordnungsgemäß endet. Beende ich eewcosimchart und starte es neu, tauchen dann mit zeitlicher Verzögerung zwei Fehlermeldungen auf. Manchmal ist es zudem notwendig eewcoSimchart 2mal zu starten.

2. Möglichkeit. Doppelklicken auf eewcosimchart_start.py. Es öffnet sich dann mit dem Programm ein cmd-Fenster, in dem etwaige Fehlermeldungen ausgegeben werden und in dem auch zum Drücken einer Taste aufgefordert wird, wenn ein Dump erstellt oder importiert worden ist.
Vorsicht: Das cmd Fenster reiht sich in diesem Fall in die Python-Programme ein und poppt nicht extra auf, wenn ein Dump erstellt oder importiert wird. Die Eingabeaufforderung ist daher etwas versteckt. (Die Eingabeaufforderung ist eingebaut, um das cmd-Fenster für den Aufruf von mysqdump und mysql offen zu halten, so dass kontroliert werden kann, ob der Aufruf der Programme Erfolg hatte.)

Zur Programmstruktur

Nun bin ich kein eingefleischter Programmierer. Mit der Programmstruktur habe ich versucht folgende Probleme zu lösen:
a) Vermeidung von Importzirkeln
b) Vermeidung von Variablengehuddel durch Nutzung von Klassen
c) und dann ein möglichst funktionaler Aufbau des Programms.
Insbesondere c) kann ich nicht abschätzen. Es ist mir nicht einsehbar, ob die entworfene Struktur tragfähig ist. a) und b) habe ich gelöst, indem ich zunächst alle Programmabläufe in Klassen gepackt habe. Dann habe ich unter jede Klasse eine Initialisierung geschrieben. Das Programm startet mit dem Modul eewcosimchart_start.py. Dieses Modul importiert zunächst alle anderen Module mit „from modulxy import *“. Dies sorgt dafür, dass die Module einzeln durchgegangen werden, und die dort gefundenen Anweisungen ausgeführt werden. Das sind die Initialisierungen der Klassen. Damit enthält der Namensraum des jeweiligen Moduls jeweils ein Objekt zu jeder Klasse. In jedem Module befinden sich ganz oben ebenfalls Importanweisungen, jetzt allerdings der Art „import modulXY“. Diese Anweisung bewirkt, dass das importierende Modul die anderen benötigten Module dem Namen

nach kennt. Diese Module werden aber mit dieser Anweisung nicht ausgeführt, so dass es zumindest auf dieser Ebene keine Importzirkel gibt.

Vom Startmodul wird dann eine Funktion im Modul hintergrund_toolbar.py aufgerufen, und von dort aus je nach Klick Funktionen andere Klassen in anderen Modulen.

Verbindung zum MySQL-Server herstellen

Die Verbindungsdaten sind in eewcoSimchart an zwei Stellen gespeichert. Der Name der Datenbank befindet sich im Dokument einstellungen.data. Er kann dort manuell geändert werden. Vorsicht: Das Wort „current_db: “inklusive einem Leerzeichen am Ende wird vom Programm verwendet, um die Zeile mit dem Datenbanknamen in dem File zu finden. Der Rest der Zeile wird dann als Name der Datenbank interpretiert.

Die anderen Verbindungsdaten befinden sich im Dokument connection.py unter der Zwischenueberschrift „# andere Verbindungsdaten initialisieren“ im oberen Teil des Dokuments.[5] Es sind dies:[6]

```
host = 'localhost'
user = 'eewcosimchart3'
passwd = 'run'
self.port = 3306
```

Um nun eine Verbindung herstellen zu können, ist eine Datenbank mit den angegebenen Verbindungsdaten im MySQL-Server und dem „current_db“ zu erstellen.

Wenn dies geschehen ist, sollten die Funktionen von eewcoSimchart zur Verfügung stehen.

MySQL-Server starten

toolbar, schema_management, mysql starten
Wenn die Verbindungsdaten von eewcoSimchart und der Datenbank im MySQL-Server übereinstimmen, lässt sich der MySQL-Server auch von eewcoSimchart starten:

Button in der toolbar unten „Schema-Management“, dann oben „mysqld starten“ klicken. Der übliche Pfad zu mysqld ist: C./Programme/MySQL/MySQL Server 5.5/bin/mysqld.exe.

Zur Programmstruktur

[5] Auch diese Eintragungen können manuell geändert werden. Dabei ist auf Erhalt der Anführungszeichen zu achten. Die automatische Anpassung bei einer Änderung der Datenbank und etwa des Passworts wird allerdings nicht unterstützt.

[6] Zu Beginn habe ich versucht, den Nutzernamen und das Passwort zu verwenden, dass automatisch für den Zugang von der Workbench aus erstellt worden ist. Dies ist mir jedoch nicht gelungen. Ich habe daher erst die Datenbank über die Workbench erstellt und dann den obigen Nutzer angelegt. Die dafür notwendigen SQL-Anweisungen sind im Dokument „user_eewcosimchart_anlegen.sql“ im Ordner docs zusammengestellt.

Die Funktion befindet sich im Modul schemata_mana.py in der Klasse cl_mysqld starten. Der Pfad wird in der Datei einstellungen.data abgelegt. Da es Probleme mit der Zeilenlänge gegeben hat, wird der Pfad in den einzelnen Stufen gespeichert.
Der eigentliche Programmaufruf geschieht dann über
self.p = subprocess.Popen([self.pfad, "--console"]).

Den Ordner für die temporären Chartdateien wählen

toolbar, schema_management, tempchartdir

Bevor eewcoSimchart den ersten Chart anzeigen kann, ist der Pfad für die temporären Ordner zu wählen, in denen die Abbildungen zwischengespeichert werden.

Der gewählte Ordner wird verwendet, um die angezeigten Charts zwischenzuspeichern. Die Charts können aus diesem Ordner zu Dokumentationszwecken auch kopiert werden.

Weitere Verwendung des Pfads???

Zur Programmstruktur
Die zugehörige Programmkomponente befindet sich im Modul connection.py in der Klasse cl_path_for_tempchartdir.

Ein Schema wählen

toolbar, schema_management, wählen

Ein Klick auf den Button „wählen“ öffnet ein Fenster, in dem die wählbaren Datenbanken angezeigt werden. Die derzeit aktive Datenbank ist vorgewählt.

Die hier gewählte Datenbank wird den Aktionen von eewcoSimchart zugrundegelegt.

Zur Programmstruktur
Die zugehörige Programmkomponente befindet sich im Modul connection.py in der Klasse cl_connection und wird mit der Funktion f_schema_aendern aufgerufen.

Ein Schema löschen

toolbar, schema_management, löschen

Zeigt die Datenbanken an, die gelöscht werden können zur Auswahl an. Die derzeit gewählte Datenbank kann nicht gewählt werden.

Zur Programmstruktur
Die zugehörige Programmkomponente befindet sich im Modul schemata_mana.py in der Klasse cl_schema_loeschen.

Ein Schema dumpen

toolbar, schema_management, dumpen

Ermöglicht die Erstellung einer Sicherungskopie der aktiven Datenbank in einem auszuwählenden Ordner und einem zu bestimmenden Namen.

Der Speicherort kann voreingestellt werden.

Zur Programmstruktur

Die zugehörige Programmkomponente befindet sich im Modul schemata_mana.py in der Klasse cl_schema_dumpen. Der dump selbst wird durch Aufruf von mysqldump.exe erstellt.
Der typische Pfad zu mysqldump.exe ist C:/Programme/MySQL/MySQL Server 5.5/bin/mysqldump.exe.

Ein Schema importieren

toolbar, schema_management, dumpen

In diesem Fenster ist es möglich, einen dump auszuwählen und in den MySQL-Server zu laden. Der Datenbankname kann dabei noch bestimmt werden.

Der Pfad zu den zu importierenden Dumps kann voreingestellt werden.

Nach dem Import wird direkt das Menü zur Bestimmung der aktiven Datenbank aufgerufen und die importierte Datenbank vorausgewählt.

Zur Programmstruktur

Die zugehörige Programmkomponente befindet sich im Modul schemata_mana.py in der Klasse cl_schema_importieren. Der Import selbst wird durch Aufruf von mysql.exe erstellt.
Der typische Pfad zu mysqldump.exe ist C:/Programme/MySQL/MySQL Server 5.5/bin/mysql.exe.

Das Simulationsmodell laufen lassen: run

toolbar, run

EewcoSimchart liest die aktuelle Modellzeit aus, gibt sie an und fragt nach der Anzahl der zu simulierenden Runden.
Die Durchlaufsequenz kann nach dem Durchrechnen eines Monats beendet werden.

Zur Programmstruktur

Die zugehörige Programmkomponente befindet sich im Modul run.py in der Klasse cl_run_pur.

Ein Run und sehen, was passiert: run & bar-chart

toolbar, run & bar-chart

EewcoSimchart liest die aktuelle Modellzeit aus, gibt sie an und fragt nach der Anzahl der zu simulierenden Runden.
Die Durchlaufsequenz kann nach dem Durchrechnen eines Monats beendet werden.
Es wird eine sogenannte Charttabelle gewählt (s.u. „Eine Charttabelle anlegen“). In einer solchen Tabelle werden bis zu 5 Variablen erfasst. Diese werden am Ende eines Monats ausgelesen und grafisch in einem Barchart dargestellt. Die y-Achse kann einzeln bestimmt werden.
Wird der Run gestoppt, können die durchlaufenden Barcharts durchblättert werden. Es ist möglich, den Run zu verlängern.

Zur Programmstruktur
Die zugehörige Programmkomponente befindet sich im Modul run_and_barchart.py.

Ein Run und sehen, was passiert: run & ts-chart

toolbar, run & ts-chart

EewcoSimchart liest die aktuelle Modellzeit aus, gibt sie an und fragt nach der Anzahl der zu simulierenden Runden.
Die Durchlaufsequenz kann nach dem Durchrechnen eines Monats beendet werden.
Es wird eine sogenannte Charttabelle gewählt (s.u. „Eine Charttabelle anlegen“). In einer solchen Tabelle werden bis zu 5 Variablen erfasst. Diese werden am Ende eines Monats ausgelesen und grafisch in einem Zeitreihenchart dargestellt. Die y-Achse kann einzeln bestimmt werden.
Wird der Run gestoppt, können die durchlaufenden Zeitreihencharts durchblättert werden. Es ist möglich, den Run zu verlängern.

Zur Programmstruktur
Die zugehörige Programmkomponente befindet sich im Modul run_and_tschart.py.

Nachsehen, was drin ist I: Bar-chart

toolbar, bar-chart

Es lässt sich eine Charttabelle (s.u. „Eine Charttabelle anlegen“) auswählen und der Barchart für einen Monat ausgeben.
Es ist möglich, mehrere Barcharts offen zu haben.

Zur Programmstruktur
Die zugehörige Programmkomponente befindet sich im Modul barchart_pur.py.

Nachsehen, was drin ist II: Ts-chart

toolbar, ts-chart

Es lässt sich eine Charttabelle auswählen (s.u. „Eine Charttabelle anlegen“) und der Zeitreihenchart für eine gewählte Zeitspanne ausgeben.
Es ist möglich, mehrere Zeitreihencharts offen zu haben.

Zur Programmstruktur
Die zugehörige Programmkomponente befindet sich im Modul tschart_pur.py.

Eine eigene Charttabelle anlegen

Voraussetzung SQL-Kenntnisse: CREATE TABLE, SELECT … INTO, INSERT VALUES, prepared statements und @user-variables, SET-statements und Verständnis einer Schleife, IF THEN END IF und NULL-Werten.

Das Auslesen und Darstellen von Werten ist für eewcoSimchart auf die folgende Weise organisiert:
Es wird eine Procedure pccharttabe_benennung in die Datenbank eingespielt. eewcoSimchart erkennt an der Präfix „pccharttabe_“, dass hier eine Procedure vorliegt, die zum Erstellen von Abbildungen verwendet wird. eewcoSimchart sucht die vorhanden pccharttabe_-Proceduren zusammen und der Benutzer kann eine davon auswählen. Die gewählte Procedure wird dann aufgerufen und der erste und letzte Monat des Zeitfensters übergeben, das für die Darstellung benötigt wird. Im Falle von Barcharts wird nur 1 Monat benötigt, so dass Anfangs- und Endwert identisch sind.

Die pccharttabe-Procedures legen dann zwei Tabellen an: eine charttabe_bennung_werte-Tabelle und eine charttabe_benennung_yachse Tabelle. In die Wertetabelle werden bis zu fünf Variablen aus den Simulationsdaten herausgelesen. In der yachse-Tabelle werden die Default-Werte für die Skalierung der y-Achsen zu den Variablen hineingeschrieben. eewcoSimchart liest dann die Werte der beiden Tabellen aus und erstellt die gewünscht Abbildung, wobei der Benutzer noch die Möglichkeit hat, die y-Achsen anders zu skalieren.

(Bemerkung zur Programmstruktur. Die Abbildung entsteht, indem mit den Werten unter Zuhilfennahme von matplotlib ein Bild erzeugt wird. Dieses Bild wird als png in einem tempchart_[nummer] genannten Ordner gespeichert. Das Bild wird dann wieder aufgerufen und in ein Bildfenster eingefügt.)

Um eine pccharttabe_-Procedure zu erstellen sind die folgenden Schritte zu durchlaufen:
1. Kopieren des Quellskripts einer vorhandenen pccharttabe-Procedure
2. Umbennen.
3. Eigene Variablen einbauen
4. die neue Procedure einspielen
5. Testen

Zu 1.

Zunächst ist ein Ordner anzulegen, indem die Skripte für die pccharttabe_Procedures untergebracht werden, mit denen eine bestimmte Datenbank untersucht werden soll. Also etwa …/modellxy [1]/simulation/charttab_skripte.

In diesen Ordner wird der Skript einer bestehenden pccharttabe_-Procedure kopiert, um so die Struktur eines pccharttabe_Procedureskripts zu übernehmen. Ein Beispielskript liegt im Ordner eewcoSimchart [1]/docs/ bei. Sie heißt sdfsdf.sql.

Zu 2.
Als erstes wird die Skriptdatei umbenannt. Dies kann ohne Einschränkung der Funktionalität frei erfolgen. Es bietet sich jedoch an, als Dateinamen die Bennennung der enthaltenen Procedure zu nehmen.

Dann ist dieser Skript zu öffnen. Beispielsweise kann dies mit Workbench geschehen. Es ginge auch mit dem Texteditor oder mit der Python-IDLE. Der Vorteil eines Programms wie Workbench ist, dass es die Syntax der Stored Procedures erkennt und so den Inhalt übersichticher darstellen kann.
Im Skript ist der Name an 8 Stellen zu ändern. Die Stellen sind jeweils durch eine Kommentarzeile kenntlich gemacht. Der alte Name ist durch den neuen zu ersetzen. Die Erkennungsteile des Namens also „pccharttabe_“ oder „charttabe_“ oder „_y_achse“ sind unverändert beizubehalten.

Zu 3.
Um eigene Werte in die Tabelle einzuspielen, sind drei Schritte zu tun, die SQL-Kenntnisse erfordern: Die Tabellenköpfe der charttabe_-Tabelle und der _y_achsen-Tabelle sind neu zu benennen. Die erste Spalte muss „runde“ bleiben. Dann können bis zu 5 weitere folgen. Für jede Spalte isteine SELECT-Abfrage zu definieren, der Wert in einer Variablen zwischenzuspeichern und sicherzustellen, dass der erhaltene Wert auch in einer Abbildung dargestellt werden kann (keine NULL-Werte).
Diese Variable dient dann in dem INSERT-Statement dazu, den Wert in die Tabelle zu schreiben.

Wenn mehr als 1 Monatswert gewünscht wird, sondern die Werte eines Zeitraums, dann wird eine Schleife entsprechend oft durchlaufen. Die SELECT-Abfragen werden jedes Mal für den jeweiligen Monat ausgeführt. Aus Gründen der Rechengeschwindigkeit werden die Werte allerdings nicht jedes Mal in die Tabelle geschrieben, sondern zunächst an eine Stringvariable angehängt, die eine INSERT-Anweisung aufbaut, mit mehreren einzufügenden Zeilen. Nachdem alle Werte herausgesucht worden sind, werden dann alle Werte mit einem einzigen INSERT-Statement in die Tabelle geschrieben. (Technik mit Prepared Statements)

Zu 4.
Skript speichern. Und dann auf den linken Blitz der Workbench drücken, der den gesamten Skript der Reihe nach ausführt. Dies spielt die Procedure in die Datenbank ein.

Zu 5.
Es empfiehlt sich dann, die Procedure zu testen, indem sie von der Workbench aus mit einer CALL pccharttabe_benennung([Anfangsmonat],[Endmonat]); aufgerufen wird. Es sollten keine Fehlermeldung aufgetaucht sein, ein blaues Ausrufezeichen ist ein gutes Zeichen. Dann mit einer SELECT * FROM charttabe_benennung_werte überprüfen, ob

die Tabelle angelegt worden ist, und die eingeschriebenen Werte den erwarteten entsprechen. Wenn ja, kann die neue pccharttabe_-Procedure mit eewcoSimchart eingesetzt werden.

Modellzeit auf 0 setzen

Mit diesem Button wird die Procedure in der Datenbank aufgerufen, die alle Werte wieder auf den Ausgangswert zurückführt.

Zur Programmstruktur
Die zugehörige Programmkomponente befindet sich im Modul hintergrund_und_toolbar.py in der Klasse cl_reset_0. Der Skript für die Reset-Prcedure befindet sich im pro_start Ordner und heißt zzz_startwerte_neu_einspielen.

Bugs, Desasters und Fancy Features

Nach einem Abbruch eine Modelldurchlaufs durch MySQL wegen eines Programmierungsfehlers, muss eewcoSimchart neu gestartet werden.

Schließen einiger Fenster über das Windows-X führt zu Abbrüchen/Fehlern, weil die aufgerufenen Funktionen nicht vollständig abgeschlossen werden.

Eventuell bei hoher Rechnerauslastung starte ich „run pur“, aber das Fenster wird nicht ausgemalt, bevor sämtliche Runden durchgerechnet sind.
-> update_idletasks() hat da schon etwas geholfen. Allerdings kommt es zu komischen weißen Feldern wo sich das Fenster zu Schemata-Mana befindet.

Wünsche und Anregungen

Wünsche und Anregungen an eewcoSim/eewcoSimchart

- Schließen der Fenster durch das Windows-X sollte ordnungsgemäß funktionieren.
- Funktions- und Strukturcheck durch erfahrene Softwareentwickler. Anlegen eines Strukturplans. (Teil davon eewcoSimchart als offenes Programm)
- Klärung, ob Versionsverwaltung notwendig. Wenn ja wie.
- Dabei auch die Fragen zu klären, ob und wie abgeleitete Modell bei Änderungen upgegraded werden können.
- Forum zu Anwendungsfragen
- Forum zur Softwareentwicklung
- Übertragung ins Englische
- eewcoSimchart verwendet das Modul os und das Modul shutil (Dateien lesen und schreiben, Ordner erstellen und löschen). Diese laufen nur auf Windows und ich meine Unix-Systemen. Gewünscht: eventl. eine Kapselung der betriebssystemspezifischen Funktionen und die Bereitstellung einer Linuxversion.
  Ebenso für das Modul PIL (Bildformat von png zu tkinter).
- Vielleicht ein Packet mit allen Programmen, die für eewcoSimchart benötigt werden mit einem Installer.
- Die Datenbank mit der die Verbindung hergestellt wird, kann gelöscht werden.
- PlugIn-Struktur, so dass neue Module nur samt Ordner in den bestehenden Ordner kopiert werden müssen. Dann könnten sie über Abfragen und Eintrag in der eigenschaften.data in Menüleisten nach individuellem Wunsch auch nachträglich eingebaut werden.
- Server-Starten Überprüfung und falls nicht läuft, Button am Programmstart präsentieren.
- Ist eigentlich so etwas wie ein Server-Stopp sinnvoll und wenn ja wie? Irgendwas mit „exit;“?
- Prüfen der voreingestellten Pfade, ob diese noch aktuell sind.
- Verbindungsmöglichkeit zu verschiedenen MySQL-Servern, insbesondere auch übers Netz (für leistungsfähige Rechenpower). Voreinspeicherung.
- Lässt sich der Fortschrittsbalken so programmieren, dass er öfter sichtbar ist? Und dass auch das Abrechen schneller funktioniert? Momentan wird die Bildmschirmanzeige vernachlässigt, sobald die Simulation gestartet wird. update idletasks() hat da nur ein bisschen was zum Besseren verändert.

- Minimierung der Charts auf dem eewcoSimchart-Hintergrund
- portables Layout
- Chart speichern und wieder öffnen können?
- Durchklicken per Tastatur
- Titel der Charts etwas aussagekräftiger
- Zusatzinformationen zu den *.png-Dateien, wie Datenbank, Variante, Chart-Procedure, Zeitraum
- Bei den Pfadänderungsfunktionen: alten Pfad beibehalten, wenn nichts gewählt
- Abbildungen, die NULL-Werte erlauben
- Eine Einrichtung zur Erstellung von Grafiken für den Druck. (Datenübergabe und Layout mit ausprobieren)
- Schließen-Möglichkeit über das Fensterkreuz für die Toolbar sinnvoll?

Reicht für andere Reiter, also momentan die Schemaorga nicht das Fensterkreuz zum schließen? done-Button kann weg.

- Feld zu Eingabe des Namens einer zu importierenden Datenbank auf 64 Zeichen setzen.
- Warnung bzw. Fehler abfangen, wenn die Kombination aus Pfad und Dumpname zu lang wird.
- Check: Funktion „wählen“. Was passiert eigentlich bei sehr vielen Datenbanken im Server? Erscheint ein Scrollbalken?
- Upperbound-limit bei Barchart von 250 passt nicht.
- permanentes Anzeigen der gewählten Datenbank
- Fenster beim Wählen der Datenbank mit einem Scrollbalken versehen.

technische Programmierungsfragen

Python

- Check for string methods in Python 3.0. ? Check what is depreceated.
- python os.system versus ecexl(),bzw. subprocess.Popen
- ware schön, wenn sich mysql auch wieder anständig schließen ließe.
- beim dumpen
  warum funktioniert Aufruf
  p2 = subprocess.Popen(„dump.cmd“)
  über Python Shell aber nicht aus Skript?
  Und warum funktioniert bei beiden:
  p2 = subprocess.call(„dump.cmd“)
- dumpen ohne Umweg über cmd-Datei
- beim Dumpen erscheint in der CMD-ausgabe vor dem „>“ eine „1“. Klappen scheint es trotzdem. Was macht die 1 da?
- Wieso wird eigentlich bei Tk button Funktionsaufruf ein event übergeben?
- bei run_and_tschart kann man vor und zurück klicken. Einmal gab es dann Fehlermeldungen, weil der Vorwärtsbutton nicht verschwand. Liegt das an der gewählten Struktur über die .after Konstruktion?

- matplotlib: Achsenskalierung auch mit Kommazahlen. Bisher nur int, sonst Fehlermeldung (run_and_tschart, line 305:
  self.charttabe_list[1][number+1] = int(neue_y_achsenwerte [number][0])

MySQL

- kann ich auch deklarierte Variable nehmen oder muss es @var sein, wenn ich Update und kumulierte Werte eintrage? (siehe mana, verw, zusagen)
- Non-transactional tables faster?? Check mit MyISAM? Auch mit den neuen InnoDB?
- nach Programmaufruf über eewcoSimchart und Fehlermeldung vom MySQL-Server die Tabellen wieder frei geben, ohne den Server herunterfahren zu müssen.
- Performance der Debugging-Hinweise mit INSERT.
- Ist es denkbar, dass Inkonsistenzen auftreten, weil MySQL intern, die Anweisungen zum Teil verzögert ausführ?
  Insbesondere bei Tabellen mit verschiedenen Engines.
  Was mit COMMMIT?
  Was mit FLUSH TABLES?
- Problem: ENUM-Type und SET SQL_SAFE_UPDATES = 0

Tücke: Ohne des safe modus gibt es keine Fehlermeldung, wenn in den WHERE-clauses beispielsweise eines UPDATEs einer ENUM-Variable ein falscher Wert zugewiesen wird. Das UPDATE wird dann einfach nicht ausgeführt.
→ gewünscht eine UPDATE-Möglichkeit ohne Key-column aber mit Überprüfung auf ENUM-Type

Zip
Zip-Ordner ist die einzige mir bekannte Möglichkeit eine Ordnerstruktur zum Download anzubieten. Es gibt von Seiten von WordPress Sicherheitsbedenken. Welche? Alternativen?

Wünsche zur Modellentwicklung und Diskussion
Darstellungwerkzeug mit dem Code, Pseudocode, Beschreibung eventuell auch Entwicklung nebeneinander gestellt werden können.
allg. Ziel: Kommunikation zwischen Modellierer und Programmierer/Umsetzung

Klarerer organisatorischer Bezug von Modellregeln und Programmierung. (Regeln und Programmierung gegenüberstellen können)

Wünsche an die externe Software
MySQL
- Fehlermeldung sollte die Procedure benennen, in der der Fehler ausgelöst wird. (weitergeleitet? noch nicht)
- Ein Modus wie SQL-SAFE-UPDATES der Updates ohne "WHERE-Clause that uses a Key-Column" erlaubt, aber nur vollständig spezifizierte Joins zulässt (Alle Variablen explizit einer Tabelle zugeordnet). (weitergeleitet? noch nicht)
- Auch ein Modus, der Fehlermeldung ergibt, wenn eine Abfrage oder eine Wertzuweisung an einen ENUM-Type eine nicht definierte Zeichenfolge beinhaltet. (weitergeleitet? noch nicht)

WORKBENCH
Löschen der Reiter zu Queryresults mit einem Klick, wenn sehr viele Ergebnisse ausgegeben worden sind.

matplotlib ?
Bisher werden nur Integerwerte in den Charttabellen Abbildungen umgesetzt. Ansonsten gibt es zum Beispiel die folgende Fehlermeldung:

```
… tschart_pur.py", line 227, in f_tschart_pur_erstellen
self.charttabe_list[1][number+1] = int(neue_y_achsenwerte [number][0])
ValueError: invalid literal for int() with base 10: '0.00'
```

Sonstige Wünsche
Version für MAC und Linux (eewcoSimChart: PIL? Installationshinweise …)

Wie werden Pythonprogramme normalerweise strukturiert?

Wie lerne ich programmieren? Materialhinweise, Kurse.

Lizenzstruktur prüfen lassen

Quellen zum Programmieren

Es gibt zur Einarbeitung in Python, matplotlib und Tkinter im Ordner eecwoSimchart/docs Linksammlungen.

Auch im MySQL-Ordner ist so ein Dokument angelegt.

Ich habe vor, ein paar Sätze über die Programmstruktur von Tkinter für Neueinsteiger zu schreiben. Bitte nachfragen.

Hinweise möchte ich insbesondere auf das Dokument: Meine_Anfaengerfehler im eewcoSim-Ordner. Hier beschreibe ich die Fehler, die jeweils die eine oder andere Stunde suchen gekostet haben. Wer also neu mit den Programmen arbeitet und schon 1, 2 Stunden nach einem Fehler sucht, kann mal einen Blick in dieses Dokument werfen.

Das ProgrammierungsKnowhow für die Stored Procedures habe ich aus:
Harrison und Steven Feuerstein: MySQL Stored Procedure Programming, Guy von O'Reilly Media (Taschenbuch - 13. April 2006).

GNU Free Documentation License

Version 1.3, 3 November 2008

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b)

permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

Verlauf der Dokumentationsbearbeitungen und die Bearbeiter

2012_01 Kapitel über die gewählte Softwarestruktur neu formuliert.
Abschnitt „Erste Einrücke eines eewco-Simulationsprogramms mit der Workbench erhalten“ neu
Abschnitt Installation neu gefasst.

2012_01 Versionstabelle eingefügt
yyy_debuggen_tabelle umbenannt in zeit_abfolge_doku und Spalte „verwendet_als_marke_von“ hinzugefügt.

2011_11 Abschnitt „Gewählte Struktur des Simulationsprogramms/Entscheidungen“ sprachlich überarbeitet, und
Parameterschnittstelle hinzugefügt, und
Handling von entscheidungsunterstützenden Tabellen formuliert.
Varianten-Abschnitt sprachlich überarbeitet.
Abschnitt über Debugging-Tabelle hinzugefügt.

2011_09 Organisation der Lizenztexte so dass zahlreiche Programmableitungen einfach zu organisieren sind. MR.

2011_09 Start eewcoSimchart: aktualisieren des Pfades zu den temporären Chartordnern. MR.

2011_08 Erstellen der ersten Version. MR.

eewcoSimchart
von Version [1] auf [2], 2012_01
Aufgrund von Fehlermeldungen bei einer Neuinstallation folgenden Änderungen durchgeführt.

tschart.py
hinzugefügt:
    from matplotlib.figure import Figure as figure
    from matplotlib import rc
    from matplotlib.pylab import *
gestrichen:
    import pylab

barchart.py
hinzugefügt:
    from matplotlib.figure import Figure as figure
    from matplotlib import rc
    from matplotlib.pylab import *
gestrichen:
    import pylab

barchart_pur.py
hinzugefügt:
    from matplotlib.figure import Figure as figure
    from matplotlib import rc
    from matplotlib.pylab import *
copy.deepcopy Anweisung für Archiverstellung auskommentiert.

run_and_barchart-py
hinzugefügt:

```
from matplotlib.figure import Figure as figure
from matplotlib import rc
from matplotlib.pylab import *
```

Die import-Anweisungen zu Tkinter, copy und PIL müssen nach der Importanweisung zu pylab stehen. Sonst werden den Namen andere Funktionen zugeordnet.

Anhänge

Ordner charttabe_beispiel
mit den .sql-Dateien „test.sql“ und „test_testen.sql“.

pdf-Meta-Elemente
Installationsanleitung und Softwarequellen für eewcoSim. Aufbau der Simulationsumgebung eewcoSim. Struktur eines eewco-Simulationsprogramms.