MARS GROUP
http://www.mars-group.org

Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

MARS DSL

ENTWICKLUNG EINER DSL ZUR KONFIGURATION EINES MULTI-AGENTEN-SIMULATIONS-SYSTEMS

Agenda

- Einführung
- Charakteristika einer DSL
- Motivation
- MARS Framework aus DSL-Sicht
- Aufgaben der MARS DSL
- Möglichkeiten der Implementierung
- User-Centered Design
- Risiken

Einführung

- MARS Framework: Ein Framework zur Erstellung von Multi-Agenten-Simulationen
- Multi-Agenten-Systeme von Natur aus komplex, da
  - jeder Agent für sich ein autonomes SW-System ist
  - Agenten untereinander mit der Umwelt wechselwirken
  - Simulationsberechnungen effizient verteilt werden müssen
- Komplexität des Systems spiegelt sich in seiner Konfiguration wieder
- End-User, die eine MAS konfigurieren müssen, sind keine Software-Entwickler

Charakteristika einer DSL

- Softwaresprache auf einer höheren Abstraktionsebene, nämlich der Anwendungsdomain
- Beispiele:
  - SQL zum Bedienen einer relationalen Datenbank
  - HTML zur Beschreibung der Struktur von Webdokumenten
- Spezialisierter Funktionsumfang im Vergleich zu GPLs
- Fokus auf die Problem-Domain steigert Effizienz bei der Problem-Abbildung

Steigerung der Produktivität der End-User

Motivation

- DSL zur Konfiguration von MAS besser geeignet als GUI?
- DSL vs. GUI
  - DSL ist schneller + besser anpassbar, erweiterbar
  - DSL-Text bietet besseren Überblick über die gesamte Konfiguration
  - DSL lässt sich kommentieren
  - DSL-Texte können untereinander ausgetauscht werden

Motivation

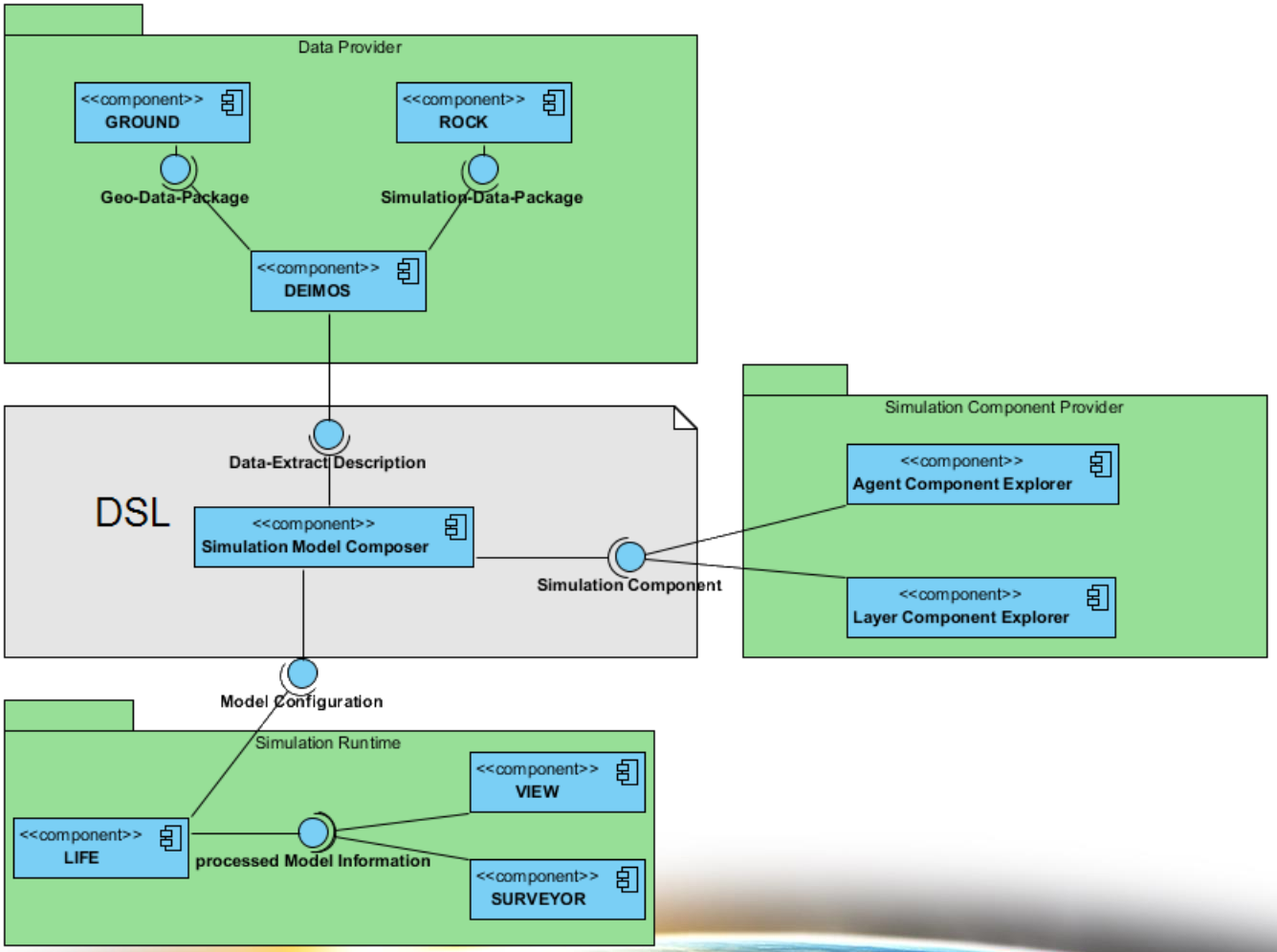
- aber: Mit DSL und Text-Editor allein erreicht man keine hohe Usability
- besondere Unterstützung von End-Usern (Nicht-Programmierer) bei der Erstellung von Quelltexten erforderlich
- Ziel: gut bedienbares intuitives Tool-Set aufbauend auf einer DSL
  - einfach zu verstehende, zu merkende und ausdrucksstarke Sprache
  - Tool-Unterstützung bei der Konstruktion und Eingabe von DSL-Statements
  - Beispiele: Autocomplete, Snippets, automatische Template-Ersetzung …

MARS Framework aus DSL-Sicht

- entwicklungstechnisches Umfeld
  - Framework, LDK und Simulationskomponenten sind in C# geschrieben
  - Userinterface der Simulation ist eine RIA (Rich Internet Application), läuft im Webbrowser
  - RIA basiert clientseitig auf AngularJS, serverseitig auf ExpressJS(NodeJS)
  - Kommunikation zwischen in C# entwickelten Systemteilen und der RIA läuft über EdgeJS

DSL und ihre Support-Tools werden in die RIA eingebettet

MARS Framework aus DSL-Sicht



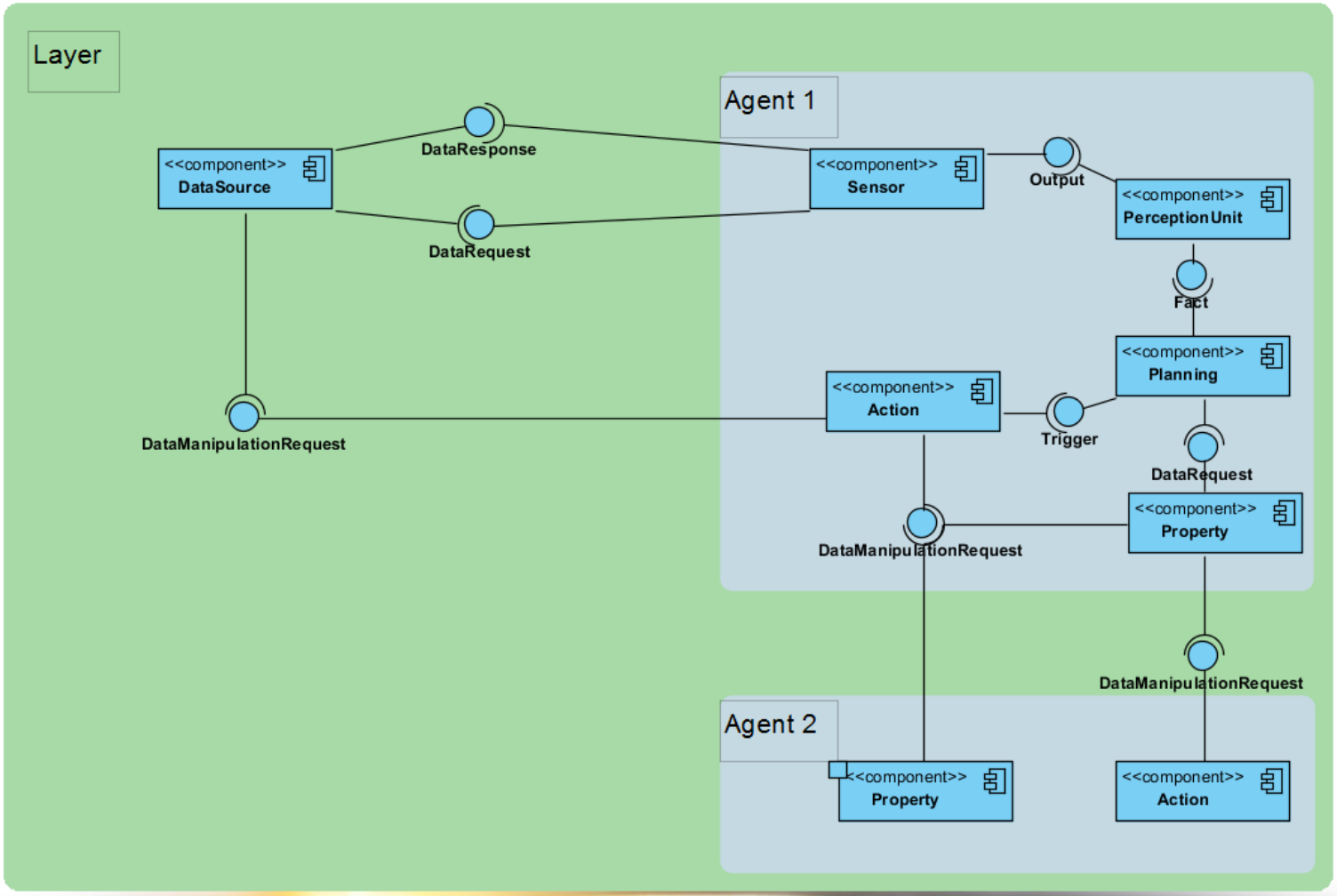
Aufgaben der MARS DSL

- Konfiguration mittels DSL
  - Zusammenfügen der Simulations-Komponenten zu funktionierenden Agenten und Layern
  - Initialisierung der Simulations-Komponenten aus den in DEIMOS zusammengestellten Daten
  - Verteilung der Agenten auf Layern
  - initialisieren und effizientes räumliches Anordnen von Agenten, die keine Grundlage in DEIMOS-Daten haben
  - zeitliche und räumliche Rahmenbedingungen der Simulation festlegen

Aufgaben der MARS DSL

- Simulations-Komponenten
  - Agenten und Layer setzen sich aus in sich geschlossenen, vorkompilierten Komponenten zusammen
  - Komponenten sind austauschbar
  - durch Hinzufügen von Komponenten ergeben sich neue Möglichkeiten der Agentengestaltung
  - neue Komponenten werden mit Hilfe des MARS LDK in C# erstellt
  - Komponenten kommunizieren über explizite Schnittstellen miteinander

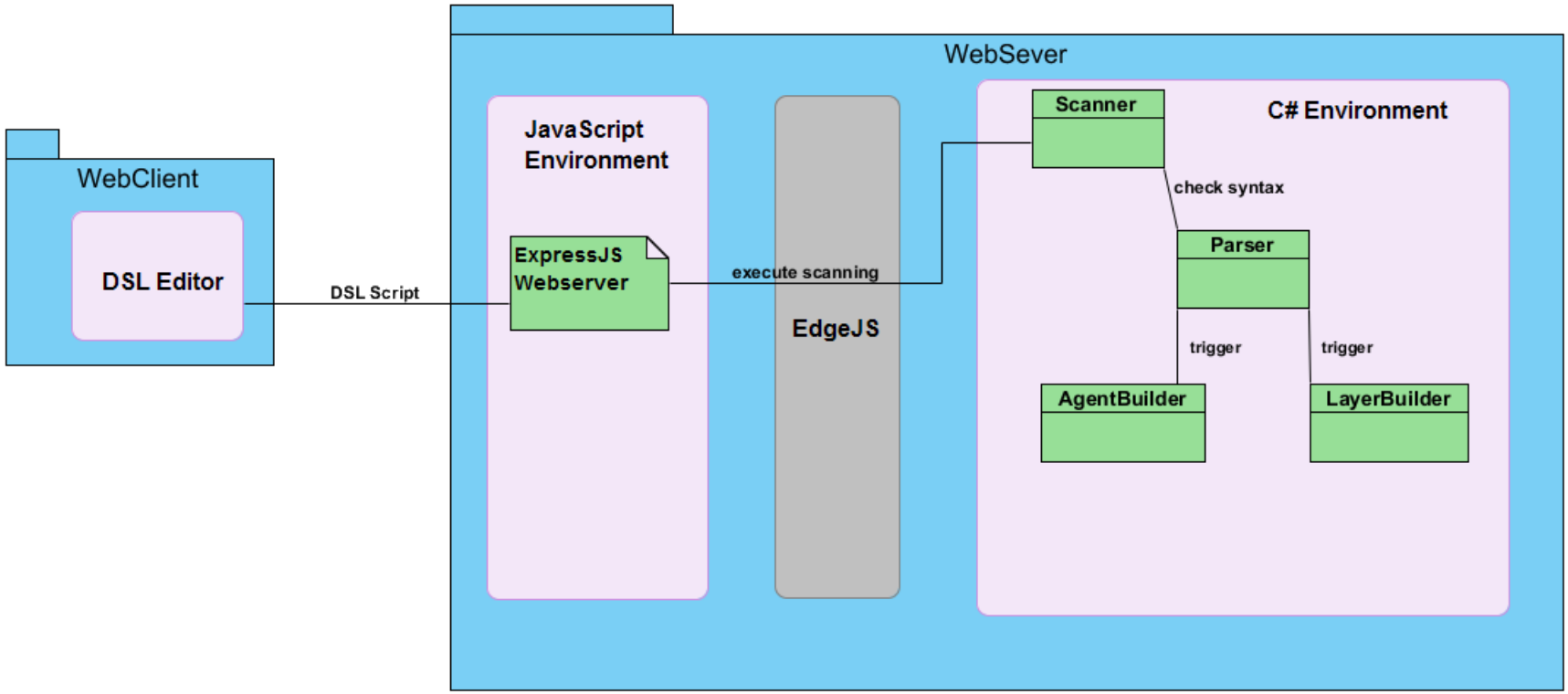
Simulations-Komponenten



Möglichkeiten der Implementierung

- Parser-Generatoren
  - basieren auf Grammatiken
  - Generator übersetzt eine Grammatik in Parser-Klassen der Zielsprache
  - erzeugte Klassen werden in die Gesamtanwendung eingebunden
  - bei validem DSL-Statement wird eine entsprechende Prozedur in der Hostsprache (hier C#) ausgeführt
  - für MARS DSL geeignete Generatoren: Coco/R, ANTLR

Einbindung eines Parser-Generators



Möglichkeiten der Implementierung

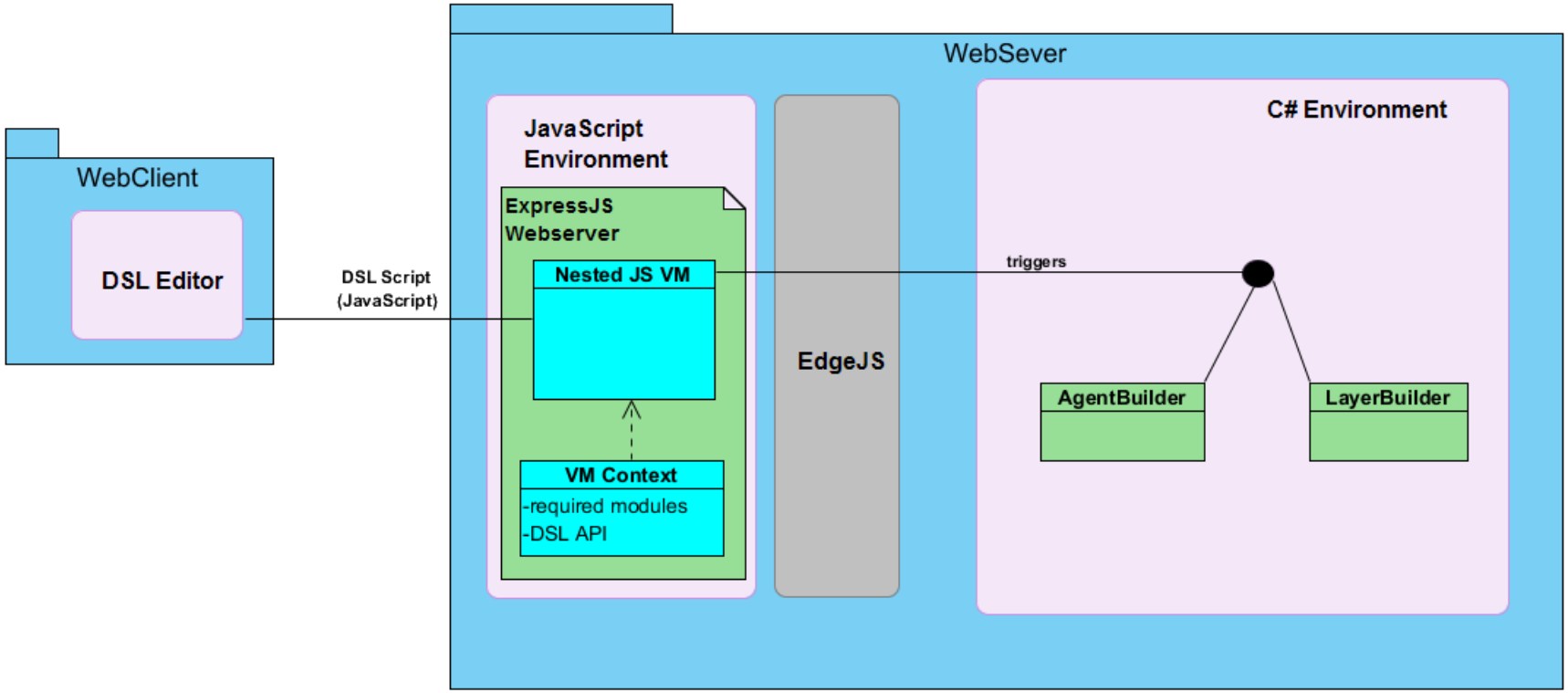
- Vorteile
  - Parser-Generator erzeugt Parser-Klassen mit anpassbarem Error-Handling
  - Freiheit bei der Gestaltung von Syntax und Semantik
- Nachteile
  - Grammatik-Syntax ist kryptisch und unübersichtlich
  - Grammatiken sind schwierig zu erweitern
  - Parser-Generierung, Einbau ins Projekt und erneutes Kompilieren nach jeder Sprachanpassung

Möglichkeiten der Implementierung

- DSL als JavaScript API
  - Aufbau einer API mit ausdruckstarken, verständlichen Methodennamen und Properties
  - JavaScript DSL Quelltext wird serverseitig kompiliert
  - Ausführung des Kompilats in eigener VM
  - eigener Laufzeitkontext für die Ausführung (Sandboxing)
  - Kontext besteht aus DSL-API und EdgeJS
  - DSL-API Funktionen als Wrapper für C# Prozeduren (Konstruktoren, Factories …)

Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Einbindung einer DSL-API



Möglichkeiten der Implementierung

- Vorteile
  - vorhandener JS Compiler kann genutzt werden, Sprachgrundlage ist bereits definiert
  - besser erweiterbar, da keine Anpassung von Grammatiken nötig
  - Build Cycle der Parser-Generatoren entfällt
- Nachteile
  - Beschränkung der Sprachgestaltung auf Methoden- und Eigenschaftenbezeichnungen
  - Fehlermeldungen zu nah an den Domain-Konzepten der Skriptsprache
  - Balance zwischen Höhe des Abstraktionslevels und Mächtigkeit der API muss gefunden werden

Möglichkeiten der Implementierung

- Tool-Support
  - Einsatz von unterstützenden Tools, die dem End-User helfen, die richtigen DSL-Statements zu erzeugen
  - Vision: Auf MARS spezialisierte Web-IDE
  - wichtige Funktionen
    - Explorer zum Überblicken von Simulations-Komponenten
    - Explorer für DSL Funktionen mit Suche, Beschreibung, Anwendungsbeispielen
    - sinnvolle und übersichtliche Repräsentation der in DEIMOS vorausgewählten Initialisierungsdaten
    - Karte zum Platzieren von Agent-Clustern

Erhöhung der DSL Usability

Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Beispiel DSL Editor

User-Centered Design

- Ziel der DSL: effizientes User-Interface zum MARS Framework

  ⇒ Hohe Produktivität der End-User

- Kriterien für Usability:
  - Effektivität
  - Effizienz
  - Zugänglichkeit
- User-Centered Design: Dialog, Tests und Evaluierung mit End-Usern während jeder Entwicklungsphase

User-Centered Design

- Vorgehen
  - Analyse der bisherigen Arbeitsweise der End-User (Tools, Workflow …) und Bewertung
  - Ergebnisse der Analyse bilden Grundlage eines DSL Entwurfs
  - Evaluierung des Entwurfs mit End-Usern. Ergebnis als Basis eines Prototypen
  - Tests und Evaluierung des Prototypen mit End-Usern
  - sukzessiver Ausbau und Verbesserung der Anwendung mit weiteren Test-Iterationen

User-Centered Design

- Basis für Evaluierung und Vergleich (bisheriges vs. neues Vorgehen der End-User)
  - subjektives Empfinden der End-User
  - Vergleich Anzahl nötiger Arbeitsschritte zum Erreichen eines Ziels
  - Vergleich des zeitlichen Aufwands
  - Vergleich der Mächtigkeit der verwendeten Tools

→ aussagekräftige Messung von Effizienz, Effektivität und Zugänglichkeit nur mit echten Simulations-Problemstellungen

Risiken

- richtiger Trade-off: Komplexität + Mächtigkeit vs. Simplizität
- kein Zuwachs oder sogar Verminderung der Produktivität der End-User
- MARS Framework noch in einem frühen Entwicklungsstadium
- u.U fehlendes Wissen über MAS bei End-Usern
- zeitlicher Aufwand des User-Centered Design

Fragen

Quellen

- Sprinkle, J., Mernik, M., Tolvanen, J.P., Spinellis, Guest editors' introduction: what kinds of nails need a domain-specific hammer ? IEEE Softw., 2009, 15–18
- Barisic, A.; Amaral, V.; Goulao, M., Usability Evaluation of Domain-Specific Languages, Quality of Information and Communications Technology (QUATIC), 2012 Eighth International Conference on the , vol., no., pp.342,347, 3-6 Sept. 2012
- Hüning, Christian; Wilmans, Jason; Feyerabend, Nils; Thiel-Clemen, Thomas MARS - A next-gen multi-agent simulation framework, Wittmann; Müller, (Ed.): Simulation in Umwelt- und Geowissenschaften, Workshop Leipzig, GI Shaker, 2014
- Janczuk T., edge.js, http:// tjanczuk.github.io/edge, Website. Abruf am 15.10.2014 20:16 Uhr
- Mernik M., Heering J., Sloane A. M. When and how to develop domain-specific languages. ACM Comput. Surv. 37, 4 2005, 316-344
- Barisic, A.; Amaral, V.; Goulao, M., How to reach a usable DSL? Moving toward a Systematic Evaluation, ECEASST, 2011